Navy Personnel Research, Studies, and Technology Division
Bureau of Naval Personnel (NPRST/PERS-1)
Millington, TN 38055-1000 NPRST-TN-05-6 June 2005

Technology Review of Multi-Agent Systems and Tools

Frank Fang
Elaine Reed
Resource Consultants, Inc.

David K. Dickason
H. James Simien
CDR Michelle L. Wulff
Navy Personnel Research, Studies, and Technology

NPRST-TN-05-6 June 2005

Technology Review of Multi-Agent Systems and Tools

Frank Fang
Elaine Reed
Resource Consultants Inc.

David K. Dickason
H. James Simien
CDR Michelle L. Wulff
Navy Personnel Research, Studies, and Technology

Reviewed and Approved by
Janet H. Spoonamore, Ph.D.
Institute for Distribution and Assignment

Released by
David L. Alderton, Ph.D.
Director

Navy Personnel Research, Studies, and Technology Department
Bureau of Naval Personnel
5720 Integrity Drive
Millington, TN 38055-1000
www.nprst.navy.mil

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 06-30-2005	**2. REPORT TYPE** Technical Report	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE Technology Review of Multi-Agent Systems and Tools	**5a. CONTRACT NUMBER** 98-D-9501/DO 7J86
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Frank Fang, Elaine Reed, David K. Dickason, H. James Simien, Michelle E. Wulff	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Navy Personnel Research, Studies, and Technology (NPRST/PERS-01) Navy Personnel Command 5720 Integrity Dr. Millington, TN 38055-1200	**8. PERFORMING ORGANIZATION REPORT NUMBER** NPRST-TN-05-6
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research 800 North Quincy Street Ballston Tower One Arlington, VA 2217-5660	**10. SPONSOR/MONITOR'S ACRONYM(S)** ONR
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

A - Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This document provided a review of current intelligent multi-agent technologies. It presents agent frameworks and agent construction toolkits available in the market today. The Navy Web-based Marketplace is a suitable environment for intelligent multi-agent technologies, where a distributed peer-to-peer process is the essential model with inter-agent communications and coordination. Reasoning with rules of thumb and heuristics are currently common practice as commands define their preferences and detailers make job assignments. This report provides recommendations for agent system development, agent framework, inference engines, and agent construction toolkits. A high-level system architecture is proposed to meet Navy Web-based Marketplace functional requirements. There are three major components of the proposal: (1) a web services-based, front-end at the presentation layer that provides friendly access for Sailors, commands, detailers, and counselors; (2) a workflow layer that provides support for business processes of the various user groups; and (3) and intelligent agent layer that provides the framework through which the agents communicate and negotiate for optimal matches between available Navy jobs and Sailors.

15. SUBJECT TERMS

Web-based marketplace, intelligent agents, multi-agent technologies

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Genni Arledge
UNCLAS	UNCLAS	UNCLAS	UNCLASS	53	**19b. TELEPHONE NUMBER** *(Include area code)* 901-874-2115 (882)

Foreword

This document provides a review of current intelligent multi-agent technologies. It presents agent frameworks and agent construction toolkits available in the market today.

The Navy Web-Based Marketplace is a suitable environment for intelligent multi-agent technologies, where a distributed peer-to-peer process is the essential model with inter-agent communications and coordination. Reasoning with rules of thumb and heuristics are currently common practice as commands define their preferences and detailers make job assignments.

This report provides recommendations for agent system development, agent framework, inference engines, and agent construction toolkits. A high-level system architecture is proposed to meet Navy Web-Based Marketplace functional requirements.

There are three major components of the proposal:

1. A web services-based front-end at the presentation layer that provides friendly access for Sailors, Commands, Detailers and Counselors.
2. A workflow layer provides support for business processes of the various user groups.
3. An intelligent agent layer provides the framework through which the agents communicate and negotiate for optimal matches between available Navy jobs and Sailors. Some background information on intelligent agent systems is included as Appendix A.

David L. Alderton, Ph.D.
Director

Executive Summary

In the current U.S. Navy Enlisted Personnel Assignment System, over 325,000 Sailors interact with approximately 300 "detailers" or assignment managers to receive future assignments. Although there should be a tripartite relationship between Sailors, commands and the Navy, detailers are usually at the center of this process,

Historically, Sailors have mistrusted the Navy's assignment process, particularly if they have been involuntarily assigned to a job in the past. Up to the present time, commands have enjoyed extremely limited input into the selection process; yet, they are held ultimately responsible for maintaining a state of personnel readiness.

Detailers currently make job assignments sequentially, based on available Sailors and requisitions (jobs), and largely in requisition priority order. However, the asynchronous availability of jobs and Sailors is inefficient and should be managed across multiple requisition cycles—a difficult task in the current system.

This research proposes to use a web-based marketplace for the distribution and assignment system. The proposed system would be demand driven and place Sailors in jobs, based on their capabilities, not job vacancies. A new assignment process methodology employing a mathematical optimization algorithm within the Web-Based Marketplace (WBM), would utilize intelligent software agents to represent both Sailors and commands. These agents would possess the requisite level of associative intelligence necessary to differentiate between multiple assignment choices, and to arrive at "best match" solutions.

This virtual marketplace may be created using an agent grid architecture, such as the Defense Advanced Research Projects Agency's (DARPA) Control of Agent Based Systems (CoABS) grid. Using global optimization methodologies, assignment slates, which are lists that consist of optimal Sailor and job matches, will be generated. The optimization objective function will provide selectable weighting coefficients to allow the Navy to maximize the net benefit to Navy readiness, while balancing Sailor and command preferences, Permanent Change of Station (PCS) costs, training costs, etc.

Using intelligent software agents, Sailors will apply for available jobs, receive descriptions of those jobs and locations, and communicate with the detailer and/or the command to request additional information, do career planning, and negotiate assignments. The Sailor agent is responsible for collecting and displaying all of the Sailor's relevant corporate data, demographic information, test scores, training and assignment history, etc. The Sailor agent will interact with the Sailor to glean personal preferences (e.g., location or platform) and special needs (e.g., disabled children, working spouse). The Sailor agent will also provide an interface to a Counselor Agent for career guidance and integrate any recommendations by the counselor agent in the search for suitable jobs. Next, an exhaustive search will be performed, followed by an evaluation of available jobs against all Sailor criteria. Finally, a list of jobs will be provided along with supporting information about each job (e.g., the job's promotion potential, available recreational amenities, school quality, etc.).

An intelligent agent will also assist the command. The command agent will be aware of the formal, minimal requirements of a job, but will also collect specific and more detailed information from each command. This additional information will focus on such areas as how a person will be used in a particular job, any desirable skills beyond the formal requirements, and any other preferences the command may have in order to support overall command personnel readiness.

This technology review document discusses the findings of a literature review, and compares existing technologies, systems, and tools that could be used to implement a Navy Web-Based Marketplace. At least two feasible multi-agent system architecture recommendations are provided.

The intended audience is all Navy personnel involved in the Web-Based Marketplace project; managers responsible for making decisions, gathering the functional requirements, determining project scope, and defining deliverables, and implementation team members (e.g., analysts, designers, developers, and testers).

Contents

Research on Technologies and Systems for Navy Detailing Requirements

The Purpose

The purpose of this technical report is threefold—to present the software agent system architectures and rule-based technologies and systems currently available; to provide a review of functionality and the underlying technology; and to recommend how the Navy's Web-Based Marketplace can be implemented using the available tools.

This effort supports the following specific objectives

- To provide a Web-based Marketplace (WBM) for Sailors to manage their preferences and access supplemental information related to their next assignment.
- To provide functionality for commands to manage their jobs and preferences for specific Sailors and skill sets.
- To provide functionality for detailers to manage applications and assignments.
- To employ multi-agent systems (MAS) technology and rule-based expert systems to create intelligent software agents for Sailors, command managers/counselors and detailers.
- To create adaptable software agents capable of increasingly complex behavior as they gain experience.

The Scope

There are many different kinds of software agent technologies and systems. Therefore, this review focused on the following two areas

- Multi-agent technologies and systems
- Rule-based technologies and systems

This decision was driven by the nature of the WBM requirements. The process of handling Sailor applications, job requisitions, counselor consultation, command preferences, and detailer slates is a very complicated process that is primarily driven by workflow. Each task requires extensive expertise and knowledge of the tasks in the field. The characteristics of the detailing process can be summarized as

Knowledge and Rule-based

There are a large number of rules, regulations, and policies that govern how assignments are made. Additionally, there are situations where the commands and detailers need to use their hands-on knowledge and established rules to determine the best assignment for Sailors to jobs. The application of preferences during the assignment decision must not only comply with established rules and Navy policy, but also with the knowledge of the commands and detailers.

Independent Agents

These independent agents (representing Sailors, counselors, commands, and detailers) perform their duties autonomously and work with other agents through collaboration, coordination, negotiation, and communication. Sailors start their search for jobs when they approach the end of their current tour and commands start to look for the best Sailor candidates for the requisitions they have in hand.

There are many agent and rule-based systems and tools available for various purposes and applications. With the Web-Based Marketplace project in mind, we focused our discussions on the following two areas.

Multi-Agent Platform and Infrastructure

This type of system is designed to provide frameworks for agents to “live” within. These systems are compared to office buildings, where companies and their employees can move in and start to work in the building. They can also be compared to communities, where people can live in the “community” with roads and other services, such as power and phone services provided. The agent platforms provide the basic infrastructure necessary to build agent communities.

Agent Construction Tools

These are the systems that are designed and developed for agent application developers to create agents with domain specific functionality. In general, the agent construction tools provide agent templates (commonly in the form of base classes) with built-in interfaces with the hosting platform and other basic functionality, such as communications.

The Approach

We have based our research on both books and information found on publicly available Internet web sites. The publications provided the technological and theoretical foundation information, and the web sites provided the latest information on systems and tools.

Although a careful and deliberate investigation was conducted, it must be noted that this is a review, and not a research paper. The intention of the review was to summarize existing technologies and to also evaluate some of the leading tools against criteria developed around the Navy’s requirements.

Summary of Agent Tools and Systems

Multi-Agent Platform and Infrastructure/Agent Construction Tools

Software agents are suitable for use in a wide variety of applications. Their versatility makes it possible to build many kinds of complex systems. However, like any other kind of technical approach, software agent systems are most appropriate for certain kinds of applications and problem domains.

Agents are best suited for use in applications that involve distributed computation or communication between components. Agent technology is well suited for use in applications where it would be particularly advantageous if the system contained components that could reason. Multi-agent systems are also suited for applications that require distributed, concurrent processing capabilities, and are particularly useful in distributed applications involving peer-to-peer processes, such as

- Process and workflow automation
- Complex planning using distributed information
- Electronic (Internet-based) commerce
- Distributed problem solving, such as asset searching and data mining
- Internet applications

Agents are already quite common in office applications. One clear example of a software agent being used in an office application is the Microsoft Office Assistant and Web searching software. Web agents can search for news that might be of interest to the user, or can help the user navigate the web. Additionally, web agents can filter and critique information so as to retrieve only what is relevant to the end-user, or act as matchmakers to bring together people with common interests.

A growing application area is in the area of bi-lateral negotiations where agents buy and sell things through interactions with other agents according to instructions from their owners. For example, Shopping Agents are created to search the Internet to locate merchandise, compare prices, and place orders using predefined shopping criteria. Job Finder Agents are built to search many sites and find employment opportunities for the user. There are also Talent Profile Agents that are created to search the Internet for resumes that satisfy certain job requirements. Professional recruiters, in the identification of new potential employees, have successfully used Talent Profile Agents.

Many kinds of agent applications can be found in automation systems. Examples include power plant control systems, aircraft autopilot and flight control systems, network management systems, and military control systems. Agent software is also found being used in the system integration arena in the form of message brokers, which are software agents able to seamlessly connect data transfer between systems using dissimilar data formats. For example, the Active Integration Suite of WebMethods can be used to integrate current and legacy software applications through either database, or Application Programming Interface (API) interfaces.

Sophisticated software agents are extremely difficult to build if they are continually constructed from scratch, as they are needed. In order to construct agents, one would need specialized skills and knowledge in a variety of areas, including agent architecture, reasoning systems, knowledge representation, and agent communication languages and protocols. If capabilities inherent in machine learning or machine planning were desired, one would need to possess skills in those areas as well. Agent construction toolkits provide a way around the issue of not possessing expertise in all of the underlying areas of software agent programming methodology. These toolkits allow software developers, without agent expertise, to quickly and easily build software agents.

There are many agent construction tools or frameworks available on the market; some of which are for academic and educational research, while others are built for commercial applications. Agent construction tools use the principle of object-oriented design and programming to encapsulate the fundamental agent building blocks, as base classes, and provide Software Development Kits (SDK) for agent developers to construct agents by providing a class library. Tools and systems are built on many different kinds of platforms and languages, including Java, C/C++, Prolog, and Python. As the following figure shows, the majority of the agent tools and systems are built using Java.

Table 1, originally created by Serenko and Detlor (2002), gives a summary of some of the popular toolkits by categories: mobile agent toolkits, multi-agent toolkits, Internet agent toolkits, and other agent toolkits.

Table 1
Summary of popular toolkits by category

Categories	Features	Language	Examples	Comments
Mobile-agent Toolkits	Mobility	Java (majority) C/C++ (<10%) Python (<10%)	Concordia Gossip Fargo	Mobile and communicate
			IBM aglets	Tool for manipulating agents
Multi-agent Toolkits	Agent interaction Communication Coordination conflict resolution	VC++ and Java	RETSINA	A framework and agent foundation class for hosting and building agents
		Java	CoABS	A middleware that integrates heterogeneous agent-based systems, objected-oriented applications and legacy systems
			AgentBuilder	Tools for building agents and hosting
			Madkit	Capable of hosting built-in agents and out-side agents
			Zeus	For developing agents and testing agents
			JADE	Framework for creating and debugging Foundation for Intelligent Physical Agents (FIPA) MAS agents.
			JADLite	Java packages for building agents
			MAST	Agents and network for knowledge interchange
Other Agent Toolkits		Java (majority) Prolog (<10%) C/C++ (<10%) Other (<16%)	FIPA-OS	Foundation for Intelligent Physical Agents (**FIPA**)
			Ascape	
Internet Agent Toolkits		Java	Microsoft Agent	Set of software services for animated presentation.
			Voyager	For creating mobile (CORBA-based) agents
			NetStepper	For creating information retrieval agents

Some construction tools are listed in Appendix B. A brief description of each tool and a link to web sites, with detailed information, has been included.

Several well-known agent development systems are summarized below.

RETSINA

The Reusable Environment for Task Structured Intelligent Network Agents (RETSINA), by the Intelligent Software Agent Lab of Carnegie Mellon University, is an extensible multi-agent system that supports communities of heterogeneous agents. The RETSINA system has been implemented under the premise that agents, in a system, should form a community of peers that engage in peer-to-peer interactions. RETSINA does not employ centralized control, rather, it implements distributed infrastructural services that facilitate the interactions between agents as opposed to managing them. A RETSINA system consists of four different types of agents: (1) interface agents interact with users, receive user input and display results; (2) task agents help users perform tasks, formulate problem-solving plans and carry out those plans by coordinating and exchanging information with other software agents; (3) information agents provide intelligent access to a heterogeneous collection of information sources; and (4) middle agents help match agents that request services with agents that provide services. RETSINA provides a development framework, with the basic agent foundation for each agent type, so agent developers can create agents that will interact with users and communicate with each other.

CoABS Grid

The Control Of Agent Based Systems (CoABS) Grid is a framework for federating heterogeneous agent systems. The CoABS Grid is middleware that integrates heterogeneous agent-based systems, object-based applications, and legacy systems. It includes a method-based, application-programming interface to register agents, advertise their capabilities, discover agents based on their capabilities, and send messages between agents. The Grid also provides a logging service to log both message traffic and other information; a security service to provide authentication, encryption, and secure communication; and event notification when agents register, deregister, or change their advertised attributes.

The CoABS grid is built using the JINI Network Technology developed by Sun Microsystems, as a specification for service discovery. The CoABS SDK provides helper and utility classes, extending from the JINI framework, for integrating agents with the framework (the grid). All necessary classes are provided including the grid configuration, directory, registration, messaging, logging, event notification, security, and grid extending classes. The framework also provides a set of utilities for agent management and status monitoring, Public Key Infrastructure (PKI) management, publishing and subscribing services.

AgentBuilder

AgentBuilder is an integrated software development tool that allows software developers, with no background in intelligent systems or intelligent agent technologies, to quickly and easily build intelligent agent based applications. AgentBuilder reduces development time and development costs, and simplifies the development of high-performance, robust agent-based systems. It contains all of the tools and capabilities for building and testing multi-agent systems.

AgentBuilder provides graphical tools for supporting all phases of the agent construction process. Programming software agents (sometimes called Agent-Oriented Programming) is accomplished by specifying beliefs, commitments, behavioral rules, and actions on the part of the agent. AgentBuilder makes it easy to create, debug, and test multi-agent systems. Software developers need only model the communication dialog between agents using the protocol tools and AgentBuilder will automatically construct the required behavioral rules to implement these conversations. This significantly reduces the amount of time required for building multi-agent systems. Tools provided by AgentBuilder include

- Organization and control of the development project using the Project Manager.
- Problem domain analysis using the Ontology Manager, which provides tools for creating ontology, concept maps, and object models.
- Automatic code generation using graphical object modeling tools.
- Special-purpose Project Accessory Classes (PACs) are available for easily adding new agent capabilities (e-mail, FTP, NNTP, etc.).
- Incorporating legacy or third-party code using Java Native Interface.
- Specifying agent behavior, including a Rule Editor for creating the agent's behavioral rules.
- Creating the flat text file (the agent definition file) that defines the agent's behavior and is executed by the run-time agent engine.
- Running the high-performance run-time agent engine
- Controlling and viewing agent execution.
- An Agency Manager for defining the agents that comprise an agency.
- Tools for defining inter-agent conversations or protocols.
- Tools for associating agent protocol roles with individual agents.
- An Agency Viewer that allows the developer to control individual agents and examine the message traffic between agents.
- Tools for automatically generating behavioral rules required for inter-agent communications.

JADE

JADE (Java Agent Development Environment) is a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through middle-ware that claims to comply with the FIPA specifications. It also simplifies implementation through a set of tools that support the debugging and deployment phase of a development project. The agent platform can be distributed across multiple machines running multiple Operating Systems and the configuration controlled through a remote Graphical User Interface (GUI). Run-time configuration can even be modified by moving agents from one machine to another, as required. The only system requirement is the Java Run Time version 1.2.

The communication architecture offers flexible and efficient messaging through which JADE creates and manages a queue of incoming Access Control List (ACL) messages that is private to each agent. Individual agents can access their queue through a combination of several modes blocking, polling, timeout, and pattern matching. The full FIPA communication model consisting of interaction protocols, envelopes, ACLs, content languages, encoding schemes, ontologies, and transport protocols have been implemented and its components were clearly delineated and fully integrated. The transport mechanism, in particular, is chameleon-like because it adapts to each situation by transparently choosing the best available protocol. Java Remote Method Invocation (RMI), event-notification, and Internet InterOperable Protocol (IIOP) are currently used, Simple Mail Transfer Protocol (SMTP), Hypertext Transfer Protocol (HTTP) and Wireless Application Protocol (WAP) are being integrated, and more protocols can be easily added. Most of the interaction protocols defined by FIPA are available and can be initiated after defining the application-dependent behaviour of each state of the protocol. Semantic Language (SL) and agent management ontology have been implemented along with support for user-defined content languages and ontologies that can be registered with agents and automatically used by the framework. JADE has also been integrated with Java Expert System Shell (JESS), a Java shell based on C Language Integrated Production system (CLIPS), in order to exploit JESS' reasoning capabilities.

JADE is being used by a number of companies and academic groups including both members and non-members of FIPA, such as British Telecom (BT), Chief of Naval Education and Training (CNET), Japan Broadcasting Corporation (NHK), Imperial College, Istituto Ricera Settore Terziario (IRST), Royal Dutch Telecom (KPN), University of Helsinki, Institut National de Recherche en Informatique et en Automatique (INRIA), ATOS and many others.

ZEUS

ZEUS provides a library of software components and tools that facilitate the rapid design, development, and deployment of agent systems. There are three sets of tools.

- The agent component library is a collection of software components that implement the functionality necessary for multi-agent systems including

    - A library of predefined coordination strategies that are represented in the form of recursive transition network graphs including several variants on contract-net, and auction protocols for more commercially oriented behavior.
    - A coordination engine that drives agent interactions by executing coordination strategies.
    - Support for several types of organizational relationships within agent societies.
    - A general purpose planning and scheduling mechanism to support goal-driven intelligent behaviors.
    - Support for agent competencies in terms of primitive actions, summary plans, forward chaining rules, and self-executing behavior scripts.
    - Representations to store and exchange information on tasks and ontology concepts.

  - An agent-to-legacy system interface to facilitate interoperability with existing software systems.
  - Full implementations of three different utility agents that provide runtime support services; agent name-to-network location resolution (Name Servers), service discovery (Facilitators), and persistent storage (Database Proxies).
- The ZEUS toolkit provides an integrated suite of editors that guide developers through the stages of comprehensive agent development methodology. During this process, developers describe how the agents interact and what tasks they will perform within their application. Among the tools provided are
  - An Ontology Editor for defining the concepts, attributes, and constraints within a domain.
  - An Agent Definition Editor for describing agents logically (e.g., their tasks, initial resources, planning abilities, etc.).
  - A Task Description Editor for describing the attributes of tasks and graphically composing summary tasks.
  - An Organization Editor for defining the organizational relationships between agents, and agents' beliefs about the abilities of other agents.
  - A Coordination Editor for selecting the set of coordination protocols with which each agent will be equipped, and the strategies that influence the agent's behavior.
- The Visualization Tools collect information on agent activity, interpret it, and display various aspects in real-time. This is our solution to the inherently difficult problem of analyzing and debugging a multi-agent system where all of the data, control, and active processes are distributed.

Cougaar

The Cognitive Agent Architecture (Cougaar) is a framework for agent developers to implement large-scale distributed agent applications with minimal consideration for the underlying architecture and infrastructure. While capable of being applied to small agent applications, the Cougaar architecture was designed as a foundation for a large-scale globally distributed system. The Cougaar model extends agent technology through the implementation of a cognitive model of human thinking and interaction. This cognitive model attempts to emulate the mental processes of perception, memory, judgment, reasoning, and decision-making.

The Cougaar architecture was developed to make heavy use of modular components. Services and capabilities within the architecture are composed of logic providers, which make adding or switching services clean and straightforward. All domain business processes, the actual behaviors of the agent, are encapsulated as one or more plug-ins, which are loaded at run-time in accordance with the configuration of the agent. Cougaar plug-ins can be dynamically loaded and unloaded at run time, as required.

Cougaar supports distributed plans, similar to a partitioned blackboard, which are interconnected, but not replicated, across the agent communities. The plans are shared among only the interested parties and efficiently managed through an implementation of publish and subscribe. A global plan is comprised of the virtual collection of the shared plan information of all the agents within the system.

Most importantly, Cougar supports the cognitive model that emulates the human ability of dynamic planning for problem solving. For a given problem, an agent will use its prior problem solving knowledge, drawn from previous experiences, relevant data, and business processes, to determine the best decomposition of specific tasks necessary to accomplish the entire task. These tasks are composed in some spatial and temporal ordering known as a workflow.

The Cougaar architecture provides complete workflow and process support functionality. It supports decompositions among tasks and constraints in addition to linkages to other tasks, triggers, and alerts. Workflows can represent alternate plans while simultaneously capturing the state of processing for all alternative paths under consideration.

Included in the Cougaar architecture is the explicit consideration for building efficient systems at large scales. Domain behaviors are encapsulated in clean code modules as plug-ins. This interface allows the domain behaviors to access services and publish information to the global plan through the plug-in API. Agent system development is further simplified within the architecture through the use of general purpose, domain independent plug-ins. For example, there are generic Structures Query Language/Java Database Connectivity (SQL/JDBC) interface plug-ins, generic Table-based Dispatcher plug-ins, generic Expert Systems plug-ins (JESS), a generic Simple Scheduler, a generic Inventory Manager, and a generic Demand Generator. Additionally, Cougaar incorporates proven commercial security technologies and a modular user interface infrastructure with an embedded web server.

The Navy Sailor Marketplace Requirement (Abstract and Architectural)

The Existing Marketplace Process

The present system relies mainly on senior enlisted personnel (detailers) to manage all enlisted assignments. This assignment system begins to work after a Sailor has finished his or her first assignment following boot camp. However, there are serious issues inherent in the current system.

- There are, at present, approximately 300 detailers located in Millington, Tennessee.
- The detailers are mainly Senior or Master Chief Petty Officers; this is a loss to the fleet of senior management skills.
- Sailors are unhappy with this system; they complain that there is too much favoritism shown by the detailers. In a case where two or more Sailors apply for a very desirable position and both are equally qualified, Sailors perceive the position will go to the person who is a friend of the detailer handling that assignment.

- The current system is somewhat a “supply driven” distribution system because Sailors are rotated between duty assignments (sea and shore) on a fixed schedule (their individual tour lengths) regardless of the availability of a replacement.
- The “first-in first-out” approach along with direct personal interaction between the Sailor and detailer is the foundation of the current enlisted assignment system. An exception is the recent addition of the Job Advertising and Selection System (JASS), which affords Sailors some flexibility in providing requisition related information.
- The overall rigidity of the assignment system requires that Sailors’ ratings match the requisition by distribution community and paygrade. Making an exception to the policy is not handled above the level of both the detailer and the Sailor, thereby requiring extensive interaction with the respective Manning Control Authority (MCA) and/or its agent, Enlisted Placement Management Center (EPMAC).
- Another major issue with the current system is the lack of anonymity on the part of the Sailor. Identities of individual Sailors in the assignment window are readily apparent to the detailers. This problem is particularly acute in the relatively small enlisted communities. Results produced by the assignment process have always been the source of skepticism, particularly on the part of those individuals who did not receive their desired assignment. Because of the lack of transparency surrounding the detailing process, Sailors have expressed concern that assignment decisions are often biased.
- Commands have little input into the requisition process. Requisition requirements are essentially driven by the information contained in the Total Force Manpower Management System (TFMMS). Therefore, commands have difficulty injecting real time requirements into the assignment process.
- Sailors must liaise directly with their detailers to provide current information about themselves. The Enlisted Assignment Information System (EAIS) has no capability, other than “duty preference,” to capture personal information that should be factored into the assignment decision.
- A lack of organizational data integrity plagues the entire assignment process. Inaccurate data permeates both the manpower and personnel corporate databases. Detailers spend an inordinate number of man-hours per month, scrubbing their requisitions for inaccurate data.

Major Characteristics of the Navy Personnel Marketplace

There are a few major characteristics that can be easily observed and identified within the Navy’s existing distribution and assignment process.

- Distributed and independent entity for handling Sailors and requisitions. Sailors represent themselves in the search for appropriate available jobs, and command managers represent their commands in seeking the best eligible candidates.
- Communication and negotiation. Sailors and detailers, as well as command managers and detailers start to discuss possible matches once a potential match is identified.

- Detailers are senior, experienced, and knowledgeable resources, who must know the requirements and skills for each requisition. There are rules that are used to relax the requirements, such as paygrade or rating communities; however, they are governed by regulations and policies.

Recommendation of Multi-Agent Development Tools

As discussed in the previous sections, intelligent multi-agent systems are complicated systems, and they normally consist of two major components—the agent environment/framework and the agents. In order for the agents to communicate with each other, the framework needs to provide basic support functions, such as messaging, agent registration or discovery, coordination and control, event handling, and logging. In other words, from an agent point of view, the agent needs to first be a "citizen" of the environment. The agent needs to have the basic communication ability to discover and exchange data with other agents throughout the framework. Therefore, all of the agents will have some degree of common functionality in the beginning.

There are many different tools or software development kits available for building agent systems, and the majority of the tools are the results of academic or government funded research projects.

A few additional comments regarding building agent-based systems

- It is complicated and expensive to construct agents from scratch without using some kind of agent construction tool.
- It is normally not recommended to build an agent framework without substantial resources and funding.
- There is no standard definition of agents, except some guidelines for agent characteristics, agent communication protocols, and languages. Therefore, agent development tools cannot be FULLY developed. In the Object Oriented (OO) term, any agent classes or objects should be extensible to allow inclusion of custom agent definition.

With these points in mind, several of the leading system/development tool candidates were selected and reviewed. All of the candidates selected for evaluation were considered to be mature and well recognized in the agent-based system field.

We specifically focused on six leading systems.

- RETSINA
- CoABS
- Cougaar
- AgentBuilder
- JADE
- ZEUS

In summary, all systems have their own special features, strengths, and weaknesses, and certainly each has been well developed and applied to certain applications. All of the systems have been discussed in various published papers and journals.

The recommendations are based on the evaluation of the previously noted agent development systems, using multiple criteria and considering the Navy's Web-Based Marketplace requirements. The goal was to identify a few alternatives that would allow NPRST

- To build a flexible, scalable, intelligent agent based system (open).
- To deliver a decent reusable solution within reasonable budget (appropriate entry with right focus).
- To meet the current needs as well as future growth (not just for now).

With these objectives in mind, the following criteria were established to measure the six systems listed above.

- Institution or company—to measure the technology credentials and academic reputation of the organization that makes the systems or tools.
- Last release date—to know if the system has been continuously supported and updated with technologies.
- Development Toolkit or SDK availability—to evaluate the system for its design and reuse capability.
- Agent support functionality—to understand the basic infrastructure support functionality.
- Scalability—to determine how scalable the system is, in terms of agents and other resources.
- Implementation Technology—to understand the technology that is used to implement the systems.
- Technological Focus—to identify the focus of the system or tool.
- Integration with Inference Engine—to identify the capability of integrating with third party inference engines for reasoning.
- Strengths and Weaknesses—to highlight the strengths and weaknesses.
- Customization and flexibility—to evaluate system flexibility and ease of customization.
- Fit for purpose—to determine how well the system fits the purpose of the Navy detailing task.

In order to form a solid recommendation, each system was evaluated using the listed criteria. The stated purpose of the Web-Based Marketplace project was "to construct an intelligent multi-agent system in which agents, representing Sailors and commands, will communicate to determine the optimal match between Sailors and jobs. All agents would be equipped with knowledge of their human counterparts and the knowledge should be easily configurable." A primary requirement was the need to be able to create software agents with reasoning capability and knowledge of both Navy human resource management, and the detailing process.

System Evaluations

RETSINA

Institution/Company

The Intelligent Software Agents Lab, Carnegie Mellon University.

Last Release Date

The distributable package was released in 2002.

Technological Focus

Agent Building and hosting. Distributed agent and service discovery.

Development Toolkit or SDK Availability

Agent Foundation Class (AFC) Kit.

Framework Support Functionality

Agent Name Server (ANS). There are several different types for human interface, tasking and planning, information retrieval, and the service matchmaker.

Scalability

Unknown.

Implementation Technology

The Kernel was originally developed in Java and later ported to Windows using Microsoft Foundation Class (MFC) and C++. The redistributed package is the AFC in VC++.

Integration with Inference Engine

It is part of the package, but open and flexible agent structure allows integration with other systems.

Strengths and Weaknesses

Distributed infrastructural services, separation of agents for GUI front-end and matchmaker middle-agents. RETSINA may not be suitable for mobile agents due to its MFC root.

Ease of Customization

It is object-oriented. Creation of customized agents is done by extending from one of several base classes. Also, the agent class library (AFC) is fully integrated into VC++ 6.0, so creating agents is assisted with VC++ wizard. The Java version may require additional technical "wrapping."

Documentation and Example Availability for Development

There are many papers and articles on the architecture and functional specifications. For developers, the AFC Developers' Guide gives excellent step-by-step examples.

Fit for Purpose

RETSINA provides sufficient encapsulation of basic agent foundation as part of its 4 base classes, and it leaves the rest open for domain specific coding. The base classes are designed as "class framework" on which agents can be derived. It is encapsulated, but not closed, to prevent customization.

Comments

Very well defined AFC for construction of any non-mobile intelligent agents. The complexity of the plumbing is well hidden and the SDK enables a quick start to creating agents.

CoABS Grid

Institution/Company

Defense Advanced Research Projects Agency (DARPA) and GlobalInfo Talk

Last Release Date

V4.2.1 released November 2002.

Technological Focus

CoABS Grid provides a framework for integration of heterogeneous objects, services, and multi-agent systems. It does not focus on any specific type of agents; instead it provides "specifications" (including helper classes) for enabling agents to become CoABS "citizens."

Development Toolkit or SDK Availability

Java helper classes and specification are provided for preparing agents for the CoABS grid.

Framework Support Functionality

Robust framework support functionality, including registration, agent discovery, communications, logging, monitoring, events, search, and security.

Scalability

The CoABS grid is scalable and tests have been conducted for tens of thousands of agents. However, it is based on the JINI network technology.

Implementation Technology

JINI network technology and Java.

Integration with Inference Engine

Not applicable since CoABS grid is not for building specific agent functionality.

Strengths and Weaknesses

Dynamic and powerful. It allows integration of almost any kind of software components and applications, including legacy systems. It is suitable for scenarios like military planning, command, execution, and combat support, where interoperability and dynamic connection are critical. It is very complicated.

Ease of Customization

Agents are built outside of CoABS, using agent construction tools. The grid is built for integrating agents and the grid can be expanded to accept agents; but the grid is not easily modified for its "hosting" functionality.

Documentation and Example Availability for Development

Several volumes of documentation on the CoABS program and the CoABS grid. There is a Grid User's Guide, where the structure of all of the helper classes is discussed.

Fit for Purpose

The CoABS grid is built as middleware for hosting agents and other software systems. It is not really a tool for building agents. Therefore, it does not fit well for the purposes discussed in this report.

Comments

CoABS grid is a very robust agent framework for federating heterogeneous services, applications, and systems across the distributed networks (LAN or WAN). Its strength is integrating different agent systems for dynamic interoperations.

Cougaar

Institution/Company

DARPA.

Last Release Date

Cougaar V10.0 released February 2003.

Technological Focus

This is a framework for agent developers to implement large-scale distributed agent applications with minimal consideration for the underlying architecture and infrastructure. The Cougaar architecture was designed as a foundation for a large-scale, globally distributed system. The Cougaar model extends agent technology through the implementation of a cognitive model of human thinking and interaction.

Development Toolkit or SDK Availability

Java packages are provided for plug-in and service development.

Framework Support Functionality

Rich framework supports functionality, including plug-in discovery, communications, task decomposition, workflow management, event notification and alerts, task planning, logging, and monitoring.

Scalability

The architecture was designed with scalability and performance as one of the primary goals. The system was tested for memory usage, CPU, network usage against a number of computers, notes, and tasks.

Implementation Technology

Java 2 Enterprise Edition (J2EE) and integration with embedded web server.

Integration with Inference Engine

There is a generic plug-in for integrating with JESS.

Strengths and Weaknesses

Dynamic and powerful platform for developing and hosting large-scale agent-based systems. In addition to the general features of agent systems, Cougaar provides support for the cognitive model and workflow management capabilities. It is very complicated.

Ease of Customization

Customization is not easy. There are many components that work together to make up the system. There is a major learning curve that the developers need to get by before effective development can start.

Documentation and Example Availability for Development

Lots of documentation is available for Cougaar, including the architecture and developer's guide. There are also training classes available for Cougaar.

Fit for Purpose

Cougaar provides much more functionality than is needed, but it certainly has tremendous potential.

Comments

Cougaar is a very powerful and robust platform for developing an enterprise level of agent applications with support of some of the latest technologies (distributed data sharing among communities, cognitive model, and workflow management) and integration with embedded web server for building web-based user interfaces.

AgentBuilder

Institution/Company

Reticular Systems Inc., California.

Last Release Date

June 2001.

Technological Focus

AgentBuilder is one of the best commercialized tools for agent construction with wizard-like graphic interface (GUI).

Development Toolkit or SDK Availability

AgentBuilder is provided as a suite for Interactive Development Environment (IDE), like Jbuilder or Visual Studio.

Framework Support Functionality

AgentBuilder offers tools for organizing and controlling the agent development project, analyzing the problem domain, specifying interaction protocols, defining the agency architecture, specifying agent behavior, viewing, and debugging agents.

Scalability

Unknown.

Implementation Technology

Implemented using Java. AgentBuilder has defined its own agent framework.

Integration with Inference Engine

Built-in rule-based reasoning module with rule editor to manage rules. The agent architecture is based on Shoham's work (1991,1993), which was based on the concept of beliefs, commitments, intentions, and behavioral rules.

Strengths and Weaknesses

Well-wrapped into an IDE suite to allow intuitive agent development; especially for people who may not have a strong agent technology background. Difficult to customize to incorporate specific custom agent logic due to the fact that agent structure is well built.

Ease of Customization

This is the weakness of this tool since it is built to allow development using an IDE. The definition of an agent is what the tool defines.

Documentation and Example Availability for Development

There is a user guide and reference manuals. The reference is more about how to use the IDE than a reference on the technical details inside the AgentBuilder (i.e., a reference that provided information for "defining" the specific agents).

Fit for Purpose

AgentBuilder has encapsulated too much so that only certain limited types of agents can be created easily; but not much customization can be done.

Comments

AgentBuilder is one of the best commercialized agent customization tools, and it is great for developing "simple" agent systems.

JADE (Java Agent Development Environment)

Institution/Company

TiLab (formerly SELT) and University of Parma, Torino, Italy

Last Release Date

March 2003.

Technological Focus

JADE is a software development framework for developing multi-agent systems and applications conforming to the Foundation for Intelligent Physical Agents (FIPA) standard. JADE SDK provides a package to develop Java agents.

Development Toolkit or SDK Availability

The JADE framework consists of two major components, the FIPA-compliant platform and the foundation class package for building agents. There are nine main packages for core services, behaviors, ACL, content, domain, GUI, protocols, FIPA, and Message Transport Protocol (MTP).

Framework Support Functionality

JADE provides a set of utilities, including remote management agent, as a graphical console for platform management and control; a dummy agent for monitoring the life cycle of agents; and Directory Facilitator (DF) as yellow pages for agent registration and discovery.

Scalability

The scalability of the JADE messaging subsystems was tested, and the results were similar to RMI when agents are distributed across the network and scales linearly.

Implementation Technology

Developed using Java, and it is recommended to use Java to develop agents using the SDK.

Integration with Inference Engine

Can be easily integrated with the Java Expert System Shell (JESS). An example was done for JESS 5.1.

Strengths and Weaknesses

JADE is a FIPA-compliant agent platform, including Agent Management System (AMS) and DF for automatic registration and naming service. However, the concern is how compatible will the agents created by JADE be with other agent hosting frameworks such as Cougaar and CoABS.

Ease of Customization

The JADE SDK provides a base Java class “agent” to be used for constructing agents.

Documentation and Example Availability for Development

There is an administrator guide and developers' guide for JADE. There are also complete examples for JESS integration agents, Java Server Pages (JSP) integration, etc.

Fit for Purpose

JADE fits quite well for what is needed for the Navy requirements. It offers enough of a base Java package.

Comments

JADE is a very well developed and recognized agent construction toolkit. Its compliance with FIPA gives JADE a tremendous advantage over other similar development toolkits. Also, its already sampled ability of integrating with JESS and JSP makes it a great candidate for web-based agent applications and systems.

Zeus

Institution/Company

British Telecommunications, United Kingdom.

Last Release Date

May 2001.

Technological Notes

Zeus provides a library of software components and tools that facilitate the rapid design, development, and deployment of agent systems.

Development Toolkit or SDK Availability

Zeus offers a library of components and visual tools for agent system development.

Framework Support Functionality

The Zeus toolkit provides an integrated suite of editors that guide developers through the stages of comprehensive agent development methodology. During this process, developers describe the agents within their application, how they interact, and the tasks they perform. It also provides visualization tools to collect information on agent activity and display various aspects in real time; this makes it easier to analyze and debug a multi-agent system within a distributed environment.

Scalability

No data available.

Implementation Technology

Implemented using standardized technology like Java, Knowledge Query and Manipulation Language (KQML), FIPA, ACL, etc.

Integration with Inference Engine

No information available.

Strengths and Weaknesses

Rich set of agent construction components and visual tools to enable rapid agent application development. There does not seem to be an active development community.

Ease of Customization

Zeus defines the agent development stages as (1) determine the candidate agents, (2) define each agent using the graphical generator tool and identify tasks, (3) use the generator tool to describe the agent relationships, (4) choose from list of prewritten coordination strategies, (5) automatically generate agent source code is, and (6) developers provide domain specific code for agent functionality. From the procedure above, customization is available only at the last step.

Documentation and Example Availability for Development

There is a technical manual discussing the Zeus toolkit architecture and its agent components. Also, the Application Realization Guide provides instructions on using the toolkits for different stages of building agents. However, there is no complete example.

Fit for Purpose

Since it seems there is no current development on this system, it would seem that it would not be appropriate for use.

Comments

The last release (code available for download) is almost two years old. Unknown if it has been updated with the latest technologies.

Comparative Analysis

Each tool was reviewed using the aforementioned criteria. A score was assigned to each appropriate criterion. The scores range from 1 to 10, where 1 is for not meeting the criteria and 10 for meeting the criteria completely. Please note the scores are used strictly to measure how well the tools are fit-for-purpose. It should be noted that this evaluation is entirely subjective and based only on the information available at the time of this evaluation.

There are features, such as strengths and weaknesses, which were not considered and/or scored here as part of the evaluation because they were judged to not be applicable with respect to the purpose stated above, but may be important for other types of projects. For example, support of heterogeneous agents is the most important strength of CoABS.

Table 2
Comparative analysis of agents

COMPARATIVE ANALYSIS CHART						
	RETSINA	**CoABS**	**Agent Builder**	**Cougaar**	**JADE**	**Zeus**
Technological Focus	Agent	Frame-work	Agent	Frame-work	Agent	Agent
Development Toolkit or SDK Availability	8	5	7	6	8	8
Framework Support Functionality	8	10	5	10	6	8
Scalability	5	7	3	9	5	4
Implementation Technology	6	7	5	7	8	6
Integration with Inference Engine	5	N/A	2	9	8	4
Strengths and Weaknesses	Not Scored	Not Scored	Not Scored	Not Scored	Not Scored	Not Scored
Ease of Customization Flexibility	8	4	2	4	7	5
Documentation and Example Availability for Development	7	5	4	4	6	4
Not Complicated Tool	6	3	5	3	5	4
Fit for Purpose	9	3	4	5	8	5
Total	62	44	37	57	61	48

Agent Construction Tool Recommendation

In general, the framework and the agent tool should not be tremendously complicated, but should be capable of being upgraded as requirements change.

The basic need is to build a system with simple agents of similar functionality, equipped with different knowledge rules.

The following are a few guidelines that were used to select the agent construction tools for development of the Web-Based Marketplace.

- The tool should provide enough plumbing foundation with basic communication capability.
- The agent built from the tool should be flexible and easily extended with domain specific processing logics.
- The agent should be capable of integration with some existing inference engines.
- The tool should not be too complicated.
- The agents created by the tool should be capable of integration into other frameworks.
- The agent and the framework should be compatible, and can be integrated with minimal effort.

Obviously, the evaluation team desires the best tool available that best fits the stated purpose. The most technologically complex, or most powerful/robust systems may not necessarily be the best fit for the development of this application.

The Java Agent Development Environment (JADE) appears to be the best candidate, but there are other factors that should be considered. One such factor is whether the development of this system is sponsored by a non-U.S. organization. If so, there may be political and licensing issues in this particular case. Therefore, in light of these and other secondary considerations, a U.S.-sponsored system was selected.

CoABS is middleware for integrating heterogeneous agents, systems, and legacy services, but it is not an agent construction tool. After careful deliberation, the recommendation was to start with a focus on agent construction, and then at a later time, consider CoABS as an upgrade option for expansion.

AgentBuilder is the best commercial tool available for building agent systems within the AgentBuilder framework. However, the agents are not easily customizable or integrated with other systems or agent frameworks.

ZEUS is a tool that combines the features of ease of use, many visual tools, and straightforward agent customization. Unfortunately, the last release of the development environment was almost two years ago. At this time it is not clear what its current status is with respect to current technologies.

Cougaar is the powerhouse of agent-based systems and additionally incorporates the cognitive model of human problem solving. Cougaar offers many features that some of the smaller systems cannot match, but the extreme richness in features is matched by an equal richness in its level of complexity. Ultimately, there is no question that it is the best candidate for the development of large-scale agent applications.

Therefore, the recommendations for the "agent construction tool" category are

- Choice 1—RETSINA by Carnegie Mellon University

  RETSINA is the fit-for-purpose. It is not extremely complicated, yet provides the basic foundation necessary to build custom agents. There is no separate, complicated, centralized agent framework. Instead, each agent owns a piece of the distributed shared environment. Also, there has been considerable work integrating RETSINA agents into other robust agent hosting frameworks, such as CoABS. Consequently, if project requirements change at some point in the future, there will be interoperability with other systems or agents. Upgrading RETSINA to a new hosting framework has previously been demonstrated.

- Choice 2—Cougaar (The Cognitive Agent Architecture)

  Cougaar represents the latest development in agent technologies. It melds software agent technologies with the latest advances from research in cognitive models for human beings. Also, there are considerable considerations for scalability and performance inherent in its design. All construction of agents is neatly encapsulated as the development of Cougaar plug-ins.

JADE is one of the best agent development environments for JAVA and examples exist that demonstrate integration between it and the Java Expert System Shell (JESS) used for agent reasoning. These capabilities seem to match quite well with the requirements of the project.

Agent Framework Recommendation

One of the critical criteria for selecting the agent framework is scalability. The number of Sailors in the U.S. Navy is estimated to be approximately 320,000. Therefore, the framework should be scalable to support as many as 320,000 agents.

The Control of Agent-Based Systems (CoABS) is a U.S. DARPA program to develop and demonstrate techniques to safely control, coordinate, and manage large systems of autonomous software agents. The CoABS project is investigating the use of agent technology to improve military command, control, communications, and intelligence gathering.

Again, the CoABS Grid is middleware that has been developed to enable the dynamic interoperability of distributed, heterogeneous objects, services, and multi-agent systems. It is adaptive and robust, with the system evolving to meet changing requirements without manual reconfiguration of the network.

The Reusable Environment for Task Structured Intelligent Network Agents (RETSINA) by Carnegie Mellon University is a multi-agent infrastructure consisting of a system of reusable agent types that can be adapted to address a variety of different domain-specific problems. Each RETSINA agent draws upon a sophisticated reasoning architecture that consists of four reusable modules for communication, planning, scheduling, and execution monitoring. Collections of RETSINA agents form an open society of reusable entities that self-organize and cooperate in response to task requirements. The agent infrastructure has been applied to a number of domains, including information retrieval and team decision-making.

The Cognitive Agent Architecture (Cougaar) provides a platform for developing and hosting large-scale agent systems, and offers features that include some of the latest agent technologies. Architecturally, Cougaar provides an easily extensible and modular model for the dynamic loading and unloading of domain specific plug-ins and applications. An added benefit is the cognitive model of Cougaar that provides an extremely capable framework for complicated problem solving and planning.

The Web-Based Marketplace operational requirements do not entail the integration of many different types of agents. Instead, it requires a few types of relatively simple software agents representing Sailors, commands, detailers, or counselors, operating in an electronic marketplace, to work out the best job assignments for a given set of Sailors and requisitions.

Given the agent construction tool recommendation and the need for fastest deployment with minimal costs, we recommend the following agent frameworks.

- Choice 1—RETSINA

  RETSINA provides sufficient framework support functionality as part of its foundation, and it does not require a centralized agent hosting facility. The RETSINA framework is a distributed architecture and allows for a quick start to the project development. It is also a proven architecture, able to be integrated upward into other more powerful frameworks such as Cougaar or CoABS.

- Choice 2—Cougaar

  Cougaar is the best and most technically innovative architecture for large-scale systems with a focus on solving complicated planning and scheduling problems. Considering the projected future growth of the Web-based Marketplace, Cougaar will be an environment to migrate to in the future. However, Cougaar is relatively complicated and requires substantial initial preparation before it would be possible to produce anything of substance.

Benefits of using RETSINA as both the framework and the agent construction tool

- Solid academic and technical support from Carnegie Mellon University
- Contains proven technologies used in real applications
- Low cost and low initial development effort to start
- Great potential for future growth and expansion with its links to Cougaar and CoABS

Basically, the solution must be flexible and reusable to allow for growth, as future needs are determined.

An Architectural Proposal

An architectural proposal for the Navy Web-Based Marketplace (WBM) consists of the following major components graphically illustrated in the following diagram.

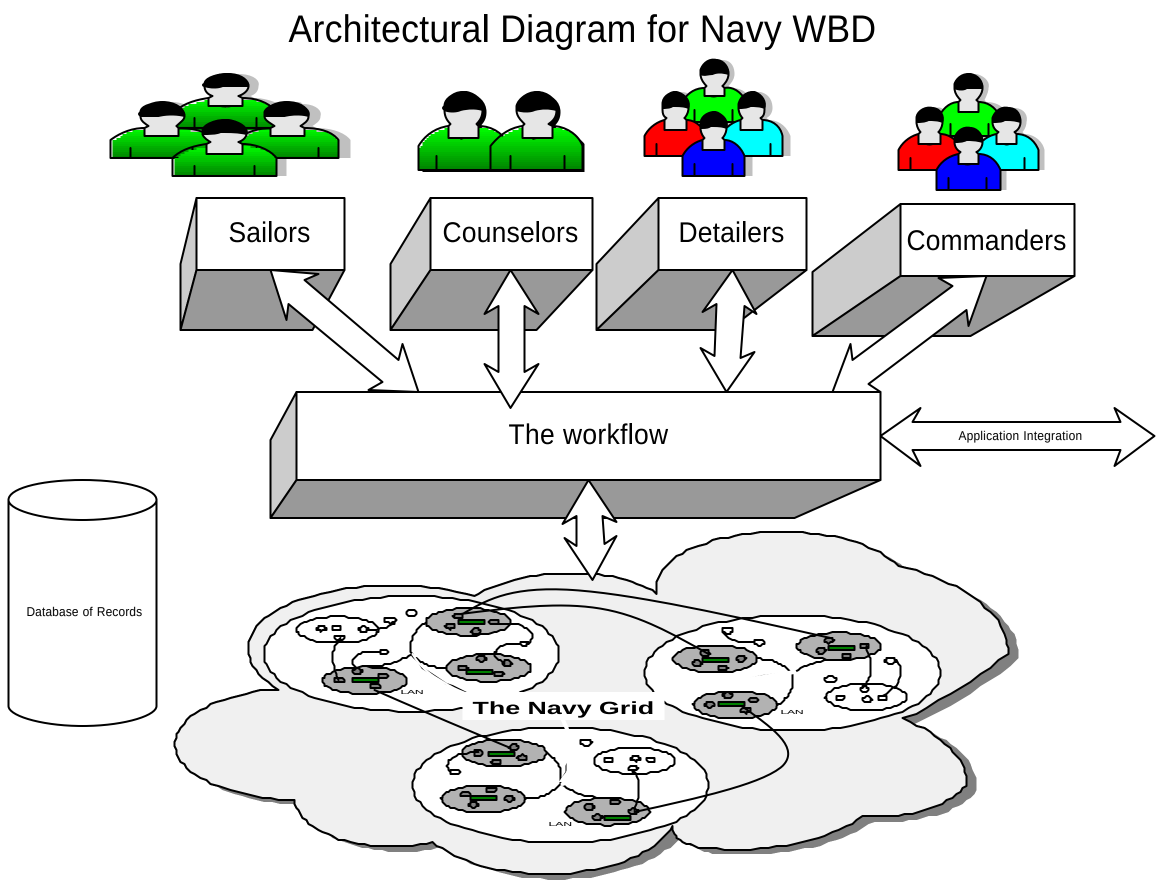

Figure 1. Architectural diagram for Navy WBD.

Web-based Presentation Layer

The presentation layer provides components for users (Sailors, counselors, command managers and detailers) of the system to perform the duties and activities related to job search and assignment.

Four major modules are listed below.

- Sailor module
- Command module
- Counselor module
- Detailer module

Workflow layer

The workflow layer provides components to implement workflows for each user group. In the workflow layer, the workflow logic provides guidance for Sailors in filling out their applications, updating their resumes, consulting with Counselors, and indicating their preferences for assignment. These workflow components would become the "preference wizard" for each user group, providing easy to follow steps for job application and assignment.

The workflow components will guide the users through the process based on their current status. For example, if a Sailor is not in an assignment window, the system should not allow him/her to fill out an application for another job. Once Sailors have applied for jobs, the process workflow will provide an automated procedure to assist the detailers in completing assignments.[1]

Intelligent Agent Layer

This intelligent agent layer hosts, or manages, pools of intelligent agents for Sailors, counselors, command managers, and detailers. Intelligent agents are software components that represent, and perform tasks for their clients (Sailors, commands, counselors or detailers). Each agent is equipped with a configurable knowledge base of rules that can be affected by regulations and policies. As the rules are adjusted, the agent knowledge base is updated thereby improving the quality of the shared agent knowledge base. The agent layer also provides utilities that are used to provide feedback to the agents for self-learning.

The intelligent agent layer consists of the agent framework and all of the intelligent agents created in the system. Agents "live," "work," "train," and "learn" within the fundamental structure, the agent framework. There are five types of intelligent agents identified as needed for the proposed system:

- Sailor agents
- Command agents
- Counselor agents
- Detailer agents
- Policy agents

The details of the modules and agents will be explained in the following sections.

In the simple figure below, the basic control-flow among the key agents have been captured to illustrate the fundamental concepts underlying the architecture.

[1] Even though the workflow layer is separated from the agents, in some agent frameworks, such as RETSINA or Cougaar, workflow processes are also implemented as agents.

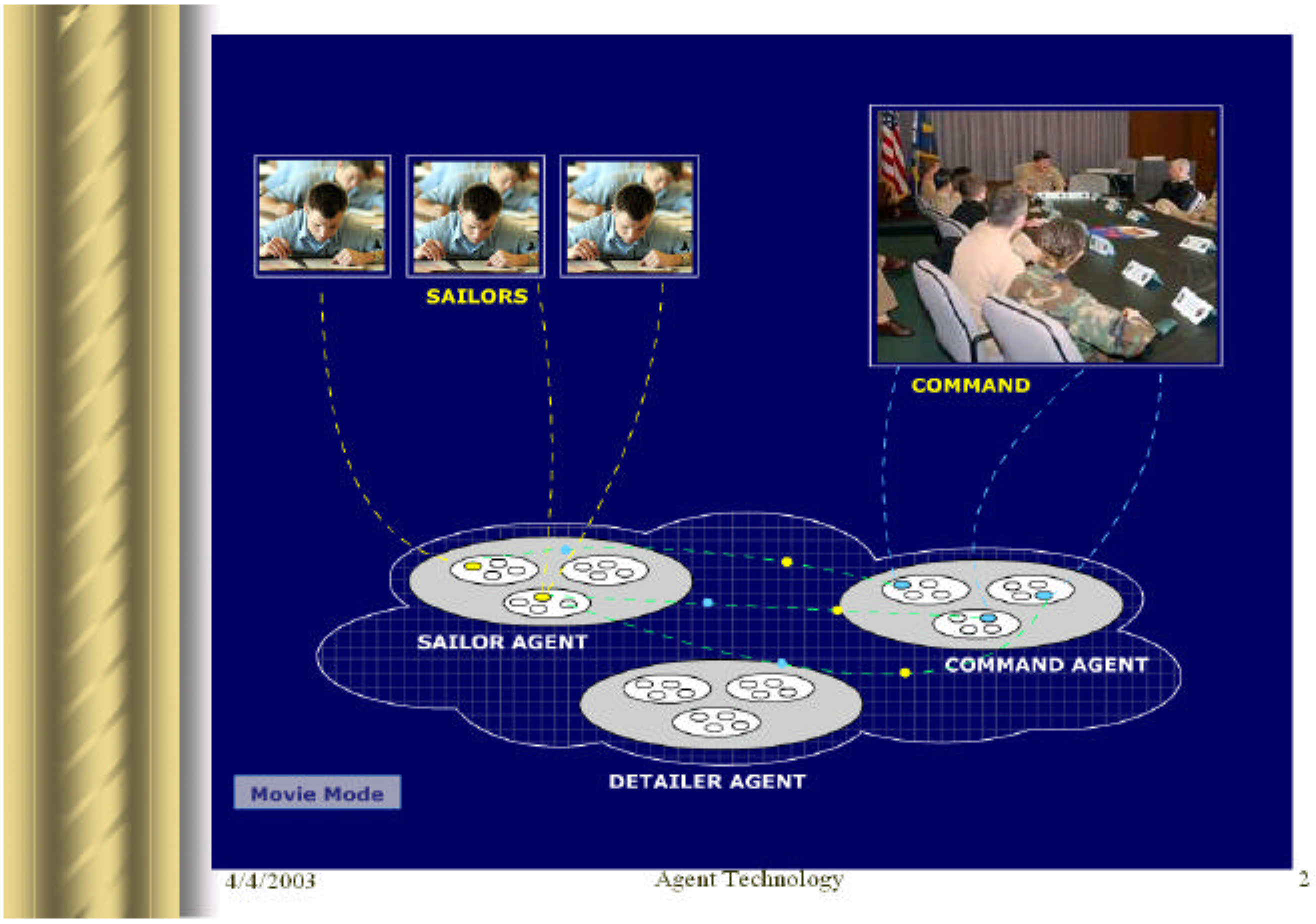

Figure 2. Key agent control-flow.

The Technology Advantages

The major characteristic of this project is the use of extremely advanced technologies. In addition to the integration of distributed data sources and web presentation, this project will be able, through the application of advanced technologies, to provide real solutions for the Navy's Sailor/personnel resource management. These are the technologies that are planned for use in the project.

- Internet, Web, and Web Service Technology
- Intelligent Multi-Agent Systems
- Rule-based Expert System Technology
- Learning and Adaptation Technology
- Enterprise Application Integration Technology
- Process Driven Workflow Management

Internet, Web, and Web-Services Technology

Internet, web, and web-services technologies will be used to enable electronic marketplace access from anywhere in the world using standard web browsers. The web services technologies provide modern display and data interfaces to legacy computing assets, which serves to streamline access, and also to enable the availability of Sailor and command-related services for other applications or operations.

Intelligent Multi-Agent Systems

Intelligent multi-agent systems technology will form the basis on which to implement the agents that will perform tasks on behalf of the Sailors, commands, and detailers. By employing intelligent agents, the complicated and time-consuming tasks of finding the best matches between applicants and jobs can be performed by the system. Agents are equipped with expertise and knowledge that is gathered from the respective subject matter experts. This expertise and knowledge is then formalized into rules that are entered into a rules engine.

Rule-based Expert System Technology

Combined with the technology of rule-based expert systems, intelligent agents separate their processing and rules engines from their knowledge base, which consists of sets of configurable and adaptable rules.

Learning and Adaptation Technology

Learning and adaptation technologies are needed to enable the agents to learn and adapt to a changing environment. Because rules and regulations change over time, it is almost impossible to attain complete knowledge of any specific situation. By incorporating learning and adaptation technologies into the software agents, the system will be much more flexible and able to meet the future manpower and personnel challenges.

Enterprise Application Integration Technology

The need to integrate with other data source systems is essential to making the electronic marketplace system work smoothly and to allow the opportunity for data sharing and collaboration with other operational organizations.

Process Driven Workflow Management Technology

In spite of the fact that workflow technology is not new, it is imperative that it be incorporated in the system in order to provide process-driven functionality for simplifying the activities of the users. As a result of process-driven flow control, process-related activities can be handled and managed automatically by the system. The efficiency and productivity of detailing would be greatly improved by taking advantage of the technology available.

Conclusion

This research effort consisted of both a review and comparison of existing technologies, systems, and tools that could be used to implement the Navy Web-Based Marketplace. To select the recommended agent construction tool and framework, six leading systems were evaluated: RETSINA, CoABS, Cougaar, AgentBuilder, JADE, and ZEUS. The selection criteria included the evaluation of institution/company credentials and reputation, latest release date, development toolkit or SDK availability, agent support functionality, scalability, implementation technology, technological focus, ability to integrate with an inference engine, strengths and weaknesses, customization and flexibility, and an evaluation of how well the system fits the stated purpose.

Of the six systems that were evaluated, RETSINA was evaluated as the top system. Cougaar ranked second for both the agent construction tool and framework that best fit the Navy's needs.

The next steps in the research will include acquisition of the agent frameworks and/or agent development toolkits, and the initiation of the following tests

- Test the fit-for-purpose of the packages, including scalability and flexibility.
- Test the compatibility between the framework and the agent construction tools.
- Test to estimate the amount of time to build agents.
- Test the integration of agents and rule-based inference engines.

References

Buchanan, B. G. & Shortliffe, E. H. (1985). *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA.

Firebaugh, M. W. (1989). *Artificial Intelligence: A Knowledge-Based Approach.* PWS-Kent, Massachusetts.

Jackson, P. (1986). *Introduction to Expert Systems.* Addison-Wesley, Reading, MA.

Serenko, A. & Detlor, B. (2002, July). *Agent Toolkits: A General Overview of the Market and an Assessment of Instructor Satisfaction With Utilizing Toolkits in the Classroom.* (Working Paper #455). McMaster University, Hamilton, Ontario.

Shoham, Y. (1993): Agent-Oriented Programming. *Artificial Intelligence*, 60(1), 51-92.

Shoham, Y. (1991): *AGENT-0: A Simple Agent Language and its Interpreter*. In Proceedings of the Ninth National Conference on Artificial Intelligence, Vol. II (pp 704-709), Anaheim, CA: MIT Press.

Appendix A:
Background on Intelligent Agent, Multi-Agent, and Rule-based Technologies

Background

The field of artificial intelligence (AI) has been one of the primary areas of research in the discipline of computer science for more than three decades. Scientists around the world have been searching for ways to create machines that are able to provide smart assistance to human beings. Developing computer systems that have the ability to see, feel, speak, listen, act, and most importantly, reason and learn, will contribute to the productivity gains of the twenty-first century. AI has been, and will continue to be, one of the most challenging areas of scientific inquiry for mankind.

Early research in Artificial Intelligence included computer software to play games like chess, pattern recognition, vision and image processing, fuzzy logic for reasoning with uncertainty, and neural networks for simulating biological learning capabilities.

Currently, the most practical AI subject is probably expert systems, which are sometimes referred to as knowledge-based or rule-based systems. An expert system is one that is capable of solving problems or giving advice in some knowledge-rich domain (Jackson, 1986). Rule-based systems have been applied in industries such as medicine and mine discovery and exploration.

Intelligent Agents

For thousands of years, human beings have been utilizing other people to perform tasks for them. If one were to thoughtfully observe their surroundings, one would notice the existence of many different types of "agents." One example would be realtors, who possess specific knowledge and expertise in the buying and selling of property, and who represent their customers in real estate transactions. In the legal system, lawyers are hired to represent clients in courts precisely because lawyers possess greater expertise than their clients in legal matters.

Additionally, human beings have been designing and creating tools to aid in the performance of useful tasks for millennia. Often, tools designed with a specific need in mind are, over time, discovered to be useful in ways that were simply unforeseen by their creators. For example, computers, which were initially developed to perform complex mathematical computations, are now used to store and retrieve vast amounts of data. Another form of agent that has received a lot of attention over the years is the robot. Robots have been successfully employed to accomplish repetitive or dangerous tasks, often in hazardous environments or under extreme conditions. Nonetheless, although many of these agents possess a high level of design sophistication, they possess little or no "intelligence."

Recently, it has become possible to create software agents that are capable of some level of reasoning. With the rapid growth of communication networks, and particularly the Internet, it is now possible to create applications that combine rule-based intelligent reasoning with distributed programs to generate software agents that are able to intelligently perform some task. Agent systems have emerged in the past five years for such sophisticated applications as data mining, military battlefield planning, robotic research, and computer gaming.

Elements of an Intelligent Agent

What is an agent? An agent is an entity that acts on another entity's behalf. While there is no generally accepted definition of an "agent," for the purposes of this discussion, an agent can be considered to be an entity with goals, actions, and domain knowledge that exists to represent humans in an electronic marketplace.

Basic agents have certain common characteristics, such as autonomy, persistence, environmental awareness, collaboration and negotiation, and communication with other agents. Intelligent agents also possess advanced abilities, such as reasoning, adaptation, and the ability to learn.

The most popular Internet software agent is a web search agent. In the case of the web search agent, the environment for carrying out a search is the World Wide Web and the computer terminal is used for interaction with the user. The agent's percepts are the words of an Hypertext Mark-up Language (HTML) document acquired from using web crawlers that continually navigate the Internet to discover new pages of information as they are added. Once a user enters the words that are to be searched for, the web search agent scans the various indexes of page content with which it is familiar to determine if it is able to find a document that matches the given search criteria. Upon determining whether a search has been successful, the agent either terminates its actions if successful, or it finds other locations to visit to continue the search.

Multi-Agent System Technology

The area of multi-agent systems (MAS) is the most prominent of the emerging subfields of AI. MAS reseach aims to provide the principles for both complex systems construction involving multiple agents, and the mechanisms for the coordination of independent agents' behaviors. Although the ability to consider coordinating behaviors of autonomous agents is a new area of inquiry, the field is rapidly advancing by building upon pre-existing work in the field of Distributed Artificial Intelligence.

Multi-agent systems differ from single-agent systems in that several agents co-exist, with each agent possessing its own goals and actions and reacting to the changes in the shared environment. In the multi-agent scenario, there may be direct interaction among agents where one agent changes the value of some environmental variables and the other agents sense the changes and take some action based upon those changes.

From an individual agent's perspective, multi-agent systems differ from single-agent systems most significantly in that the environmental dynamics can be influenced by other agents. In addition to the uncertainty that may be inherent in a particular domain, other agents intentionally affect the environment in unpredictable ways. Thus, all multi-agent systems can be viewed as existing dynamic environments.

Parallelism is achieved by assigning different tasks or abilities to different agents. Robustness is a benefit of multi-agent systems that have redundant agents due to the fact that if control and responsibilities are sufficiently shared among agents, the system can tolerate failures of one or more of the agents. Application domains that must degrade gracefully are in particular need of this feature of multi-agent systems. If a single entity, processor or agent, controls everything, then there is a single point of failure. Although a MAS need not be implemented on multiple processors to protect against failure, its agents should be distributed across several machines.

Another benefit of multi-agent systems is their scalability. Since they are inherently modular, it is easier, in theory, to add new agents to a multi-agent system than it is to add new capabilities to a monolithic system. Systems with capabilities and parameters that must change frequently adaptability can also benefit from this advantage of multi-agent systems.

Figure A-1 illustrates the view that each agent is both part of the environment and modeled as a separate entity. Agents model each other's goals and actions; they may also interact directly (communicate).There may be any number of agents, with different degrees of heterogeneity, and with or without the ability to communicate directly.

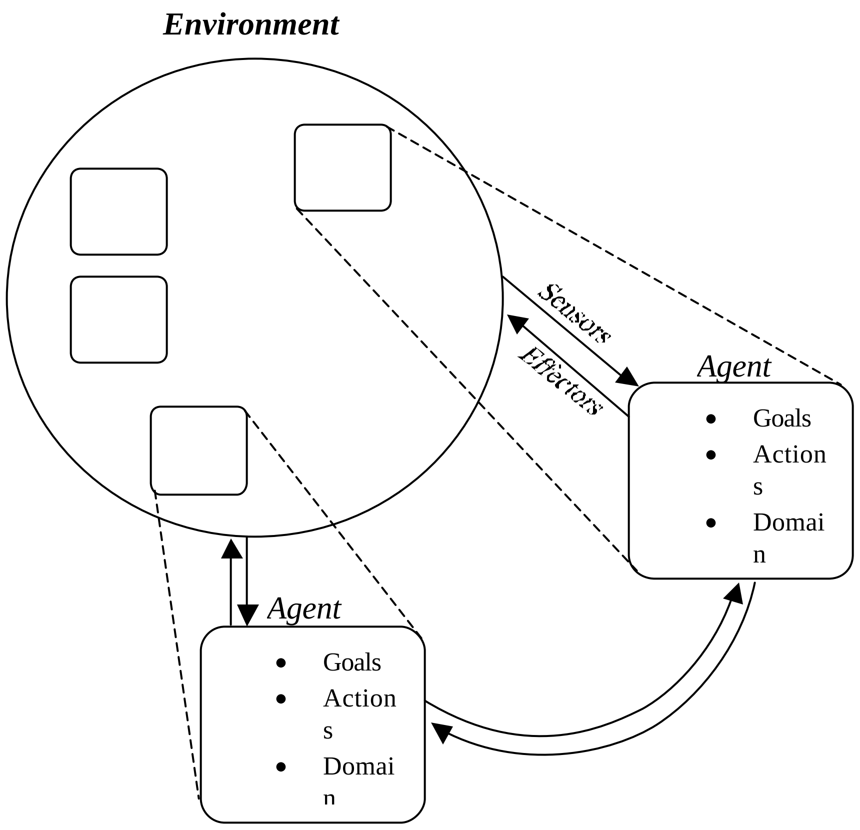

Figure A-1. The fully general multi-agent scenario.

Rule-based and Knowledge-based Expert Systems

Conventional programming languages, such as Basic and C/C++, are designed and optimized for the manipulation of data at a fairly low level in comparison with the way people approach the manipulation of data. Humans often solve complex problems using mental models, called heuristics that are based on applying rules to patterns. The human approach is one that is not well suited for implementation in conventional computer programming languages because, although abstract information can be modeled in these languages, considerable programming effort is required to transform the information into a format usable with current programming paradigms.

One of the results of artificial intelligence research has been the development of techniques, which allow modeling of information at higher levels of abstraction. These techniques are embodied in languages or tools, which allow programs to be built that closely resemble human logic in their implementation, and are therefore easier to develop and maintain. These programs, which emulate human expertise in well-defined problem domains, are called expert systems. The availability of expert system tools, such as C Language Integrated Production System (CLIPS), has greatly reduced the effort and cost involved in developing an expert system.

Rule-based programming is one of the most commonly used techniques for developing expert systems. In this programming paradigm, rules are used to represent heuristics, or "rules of thumb," which specify a set of actions to be performed for a given situation. A rule is composed of an if-portion and a then-portion. The if-portion of a rule is a series of patterns, which specify the facts (or data) that cause the rule to be applicable.

Pattern matching is the process of matching facts to patterns. An expert system tool provides a mechanism, called the inference engine, which automatically matches facts against patterns and determines which rules are applicable. The if-portion of a rule can actually be thought of as the whenever portion of a rule, since pattern matching always occurs whenever changes are made to facts. The then-portion of a rule is the set of actions to be executed when the rule is applicable. Actions of applicable rules are executed when the inference engine is instructed to begin execution. Once execution begins, the inference engine selects a rule and then the actions of the selected rule are executed which may affect the list of applicable rules by adding or removing facts. The inference engine then selects another rule and executes its actions, thereby repeating the cycle until no applicable rules remain.

Inference Engine

The most important part of an expert system is the inference engine, which is responsible for resolving the many IF-THEN statements that are found in all expert systems. These statements are often rather complicated and inter-dependant, and form the basis for the rules an expert system follows. An expert system cannot function without an inference engine.

The inference engine is the generic control mechanism that applies the axiomatic knowledge, present in the knowledge base, to the task-specific data to arrive at some conclusion. This is the second key component of all expert systems. Having a knowledge base alone is not of much use if there are no facilities for navigating through, and manipulating, the knowledge in order to draw conclusions.

Again, the knowledge base of rules and the inference engine are the two key components of any rule-based expert system. The rules are extracted from the domain experts and represented in a format that can be understood by the inference engine.

There are many inference engines or expert system shells available in the market and Appendix C lists some of the most popular. Many of the inference engines, such as CLIPS and Official Production System 5 (OPS5), are available free and can be downloaded on the Internet; others are built for commercial applications.

Reasoning Approaches—Forward and Backward Chaining

Forward chaining is a reasoning approach that finds every conclusion possible based on a given set of premises. Forward chaining is used primarily for making a diagnosis where all possible outcomes of a given circumstance, based on given input, are under consideration. For all other purposes, forward chaining is only feasible when the number of possible outcomes is small; otherwise, it would be more advantageous to use backward chaining. When many conclusions exist, forward chaining becomes too inefficient and time consuming to be of any use.

On the other hand, by assuming a desired outcome, backward chaining resolves given problems by checking to see if the rules are met. With this approach, an expert system would need to do fewer operations than would be performed using forward chaining. Backward chaining eliminates having to solve for every possible outcome based on a given set of rules.

The Knowledge Base and the Rules

The knowledge base of an expert system is the data store of knowledge of the expert system; it stores the rules that specify how the inference engine will work. Rules are the special representation of knowledge in the forms of premises and conclusions. All of the knowledge base rules, taken together, form the "expertise" inherent in the expert system. The rule base is the set of the "rules of thumb" used by the expert system to solve problems.

Learning and Adaptation—Self-Improvement

Machine learning is the area of AI that focuses on developing principles and techniques to automate the acquisition of knowledge. Some machine learning methods can dramatically reduce the cost of developing knowledge-based software by extracting knowledge directly from existing databases. Other machine learning methods enable software systems to improve their performance, over time, with minimal human intervention. These approaches are expected to enable the development of effective software for autonomous systems that must operate in poorly understood environments.

Adaptation and learning in MAS establishes a relatively new, but significant, topic in AI. The MAS environment is full of "dynamics," that might be well defined, or changed and updated by participating agents. Therefore, MAS are typically very complex and demonstrate "hard to specify" behavior. It is broadly agreed in the AI community that the need exists to endow these systems with the ability to adapt and learn.

Web Services

Web services are self-contained, modular applications that can be described, published, located, and invoked over a network, including the Internet.

A web services architecture is the logical evolution of object-oriented analysis and design coupled with the logical evolution of components geared towards the architecture, design, implementation, and deployment of e-business solutions. As in object-oriented systems, some of the fundamental concepts in web-services are encapsulation, message passing, dynamic binding, and service description and querying.

The Web Services architecture describes the principles behind the next generation of e-business architectures, presenting a logical evolution from object-oriented systems to systems of services. Web services systems promote significant decoupling and dynamic binding of components. Because all components in a web services system are services, they conveniently encapsulate program behavior and publish a messaging Application Programming Interface (API) so that other collaborating components on the network are able to access common components. Under this approach, services are marshaled by applications using service discovery to dynamically bind collaborations. Web services reflect a new service-oriented architectural approach to application development, based on the notion of building applications by discovering and orchestrating network-available services, or just-in-time integration of applications.

At runtime, application execution is a matter of translating the collaborator requirements into input for a discovery mechanism, locating a collaborator capable of providing the right service, and orchestrating messages to collaborators to invoke their services. These new applications become services, thus creating aggregated services that are available for discovery and collaboration.

Some of the benefits that web services provide include

1. Enabling just-in-time integration—dynamic service discovery and invocation (publish, find, bind), along with message-oriented collaboration, yield applications with looser coupling, which enables just-in-time integration of new applications and services. This in turn allows the development of systems that are self-configuring, adaptive, and robust.
2. Reducing complexity by encapsulation—all components in web services are services. What is important is the type of behavior a service provides, not how it is implemented. Encapsulation is the key to coping with system complexity because application designers are freed from having to worry about the implementation details of the services they are invoking in their applications.
3. Enabling interoperability of legacy applications—by allowing legacy applications to be wrapped and exposed as services, the web-services architecture easily permits new interoperability between applications. Additionally, security, middleware, and communications technologies can also be wrapped to participate in a web-service as environmental prerequisites.

Appendix B:
Multi-Agent Systems and Tools

The Academic and Research Systems and Tools

Control of Agent-Based Systems (CoABS)

Language

Java and Jini

Type

Multi-agent infrastructure

Organization

DARPA; GlobalinfoTek, Inc.

Release Notes

Well-tested with 10,000+ agents registered in grid

Functional Description

Control of Agent-Based Systems (CoABS) is a program of the U.S. Defense Advanced Research Projects Agency (DARPA) to develop and demonstrate techniques to safely control, coordinate, and manage large systems of autonomous software agents. CoABS is investigating the use of agent technology to improve military command, control, communication, and intelligence gathering.

The CoABS grid is middleware that has been developed to enable the dynamic interoperability of distributed, heterogeneous objects, services, and multi-agent systems. It is being used to produce militarily relevant technical integration experiments, where legacy systems and multi-agent systems developed by CoABS researchers are integrated to solve real-world problems. The CoABS grid features flexible run-time communications and dynamic registration and discovery of relevant participants. It is adaptive and robust, with the system evolving to meet changing requirements without reconfiguring the network. A significant effort is being made to transition and deploy the CoABS Grid and associated agent technologies to the operational user community.

Agent Building Shell (ABS)

Language

COOrdination Language (COOL)

Type

Agent architecture

Organization

Enterprise Integration Laboratory, University of Toronto

URL

Agent Building Shell Site

Functional Description

An Agent Building Shell that provides several reusable layers of languages and services for building agent systems; coordination and communication languages, description logic-based knowledge management, cooperative information distribution, organization modeling, and conflict management. The approach is being used to develop multi-agent applications in the area of manufacturing enterprise supply chain integration.

Agent Factory

Language

Visual Works

Type

Agent prototyping environment

Organization

University College, Dublin, Ireland

URL

Agent Factory Site

Functional Description

Agent Factory is an agent prototyping environment. It features the following

- Written in Smalltalk-80 (supports Mac, PC, UNIX)
- Distribution through TCP/IP
- Web interface provided
- Agents are reactive, social, pro-active, autonomous, and intentional
- Agents may be mobile
- Agent model allows configuration of
  - Mental state architecture (the developer is free to construct a mental state architecture; however, the inclusion of beliefs and commitments is expected)
  - Agent communication language (default is Teanga ACL, but KQML and FIPA ACL are supported)
  - Intentional reasoner (commitment management system)
- Agent designs require initial mental state, preceptors, actuators.

Tool Command Language (Tcl)

Language

Tcl

Type

Mobile agents

Organization

Dartmouth University

URL

Agent Tcl Site

Functional Description

Agent Tcl is a tool for developing transportable agent systems. The transportable agents are created using the Tool Command Language (Tcl). Tcl is an embeddable scripting language that is highly portable, highly popular, and freely available. The agents migrate from machine to machine using the jump command. Execution resumes on the destination machine at the statement immediately after the jump is completed. Modifications to the Tcl core allow the capture of the complete internal state of an executing script. Migrating agents are encrypted and authenticated using Pretty Good Privacy (PGP).

Access restrictions are imposed on the agent, based on its authenticated identity. Safe Tcl enforces the access restrictions. In addition to migration, Agent Tcl supports message passing. Agents can clone themselves and the system provides rudimentary security features. Each agent on a particular machine has a unique integer ID and a unique symbolic name. Agents specify a recipient agent by specifying the recipient's machine and either the recipient's integer ID or the recipient's symbolic name. The research project is addressing issues involving debugging, privacy, security, mobile agent management, networking resources and performance. Agent Tcl has two components: a modified Tcl interpreter that executes Tcl agents and a server, which runs on every machine that can receive a transportable agent.

agentTool

Language

Java

Type

Toolkit

Organization

Air Force Institute of Techno logy

URL

AgentTool site

Functional Description

The goal of agentTool is to allow agent system designers to formally specify the required structure and behavior of a multi-agent system and semi-automatically synthesize multi-agent systems that meet those requirements. High-level system behavior is defined graphically using multi-agent systems engineering methodology.

Architecture type-based Development Environment (ADE)

Language

Java

Type

Platform and application independent agent development

Organization

University of Potsdam

URL

Architecture type-based Development Environment (ADE) site

Functional Description

Software agents and agent systems are modeled in an object-oriented manner, using the latest findings of software architecture. This approach describes software architecture as a system of components with interactions realized via connectors. With the architecture-based methodology, theoretical fundamentals for the implementation of computer-independent platforms for agent programming have been found. Existing platforms, such as IBM Aglets or Objectspace Voyager, do not support an architecture-based approach and environment when developing agent application systems.

As a result, these platforms miss the explicit modeling of agent interactions, or allow only a very limited number of property classes. The advantages of a substantiated methodology of an architecture type-based approach are lost while transforming them onto existing platforms. To map the mental models and systems onto agent platforms, an architecture type-based agent development environment for the modeling, analysis, and construction of agent application systems for various agent platforms called Architecture type-based Development Environment (ADE) is being developed.

Ascape

Language

Java

Type

Software framework

Organization

The Brookings Institution

URL

Ascape site

Functional Description

Ascape is a software framework for developing and analyzing agent-based models. In Ascape, agent objects exist within scapes (i.e., collections of agents such as arrays and lattices). Since these scapes are themselves agents, typical Ascape models are made up of “collections of collections” of agents. Scapes provide a context for agent interaction and sets of rules that govern agent behavior. Ascape manages graphical views and collection of statistics for scapes, and provides mechanisms for controlling and altering parameters for scape models.

Bee-gent

Language

Java

Type

Software development framework

Organization

Toshiba Corporation, Systems and Software Research Laboratories

URL

Bee-gent site

Functional Description

Bee-gent is a new type of software development framework in that it is a 100 percent pure agent system. Unlike other systems, which make only some use of agents, Bee-gent completely "agentifies" the communication that takes place between software applications. The applications become agents and agents carry all messages. Thus, Bee-gent allows developers to build flexible open distributed systems that make optimal use of existing applications. The Bee-gent framework is comprised of two types of agents. "Agent Wrappers" are used to agentify existing applications, while "Mediation Agents" support inter-application coordination by handling all communications. The mediation agents move from the site of an application to another where they interact with the agent wrappers. The agent wrappers themselves manage the states of the applications they are wrapped around, invoking them when necessary. Thus, inter-application coordination is handled by the agent wrappers generating and receiving requests, which are transported around by the mediation agents. The mediation agents do more than just transport the messages; they are able to respond to the nature of the request to determine the best course of action.

Bond Distributed Object System

Language

Java

Type

Agent framework

Organization

Purdue University

URL

Bond Distributed Object System site

Functional Description

The Bond Distributed Object System provides a message oriented middleware environment for developing distributed applications. Bond uses the KQML language for object communication. The agent framework of the Bond system simplifies the task of developing agents by allowing the programmer to concentrate on the specific strategies of a new agent. Bond agents have the intrinsic capability to be controlled remotely and to cooperate with each other. The task of an application programmer is limited to specifying the agenda, the finite state machine of the agent, and the strategies associated with each state.

Cable

Language

C++ and ADL

Type

System architecture

Organization

Logica

URL

Cable site

Functional Description

Cable is a generic system architecture developed by Logica as part of the GRACE Consortium. Cable can be used to develop and execute distributed applications that are based on the metaphor of multiple, cooperating intelligent agents. Cable provides the user with an Agent Definition Language (ADL) for defining agents, and a parser known as the Scribe, for compiling agent definitions written in ADL into agent applications. Agents are developed using ADL and C++. ADL allows developers to use Cable without worrying about underlying detail, providing a language with a level of abstraction close to that of the agents with which an application is designed. Inter-agent communication over a local area network is handled using ORBIX, an implementation of the CORBA 2.0 standard.

Distributed Environment Centered Agent Framework (DECAF)

Language

Java

Type

Agent framework

Organization

University of Delaware

URL

DECAF Agent Framework site

Functional Description

DECAF provides a platform for extremely rapid development of agents. This is accomplished by building an operating environment that provides an interface, internal agent scheduling, and monitoring in a fashion similar to operating system primitives. The agent developer does not need knowledge of any of this structure and can thus focus on development of the agent itself. The basic DECAF architecture has been built using the Java programming language. The rapid prototyping has been tested by development of approximately 15 agents over a 2-week period by a recent class.

Hive

Language

Java

Type

Toolkit for building distributed systems

Organization

The Media Lab, Massachusetts Institute of Technology

Functional Description

Hive is a Java software platform for creating distributed applications. Using Hive, programmers can easily create systems that connect and use data from all over the Internet. At its heart, Hive is an environment for distributed agents to live, communicating and moving to fulfill applications.

Java Agent Development Framework (JADE)

Language

Java

Type

Multi-agent framework

Organization

CSELT S.p.A, University of Parma

URL

JADE site

Functional Description

JADE is a software framework to develop agent-based applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems. The goal is to simplify the development while ensuring standard compliance through a comprehensive set of system services and agents. JADE can then be considered an agent middleware that implements an agent platform and a development framework. It deals with all those aspects that are not peculiar to the agent internals and that are independent of the applications, such as message transport, encoding and parsing, or agent life cycle. All agent communication is performed through message passing, where FIPA ACL is the language to represent messages.

JATLite

Language

Java

Type

Java packages for multi-agents

Organization

Stanford University

URL

JATLite site

Functional Description

JATLite is a set of Java packages that make it easy to build multi-agent systems using Java. JATLite provides a basic infrastructure in which agents register with an Agent Message Router facilitator using a name and password, connect/disconnect from the Internet, send and receive messages, transfer files, and invoke other programs or actions on the various computers on which they are running. JATLite facilitates construction of agents that send and receive messages using the emerging standard communications language, KQML (see http://www.cs.umbc.edu/kqml/ for the current KQML standard). The communications are built on open Internet standards, TCP/IP, SMTP, and FTP.

Java Agent Framework (JAF)

Language

Java

Type

Agent framework

Organization

University of Massachusetts

URL

Java Agent Framework (JAF) site

Functional Description

JAF is a component-based agent architecture, which allows the designer to use and build software agents. It consists of a design methodology and a set of reusable components, which together allow you to design, extend, and integrate components into a cohesive whole. The framework includes structures and components for task modeling, planning, scheduling, method execution, and communication, among others. The available distribution includes examples of both a stand-alone agent and one designed to function within the MASS simulation environment.

Java Intelligent Agents Componentware (JIAC)

Language

Java

Type

Agent framework

Organization

Technische Universitat Berlin

URL

JIAC site

Functional Description

Java Intelligent Agents Componentware (JIAC) is an open, scalable, and universal agent architecture. Implemented in Java, it offers component-based mobile agents, as well as providing support for creating e-commerce applications and distributed telecommunication services.

Reusable Environment for Task Structured Intelligent Network Agents (RETSINA)

Language

Type

Organization

Carnegie Mellon University

Functional Description

The RETSINA multi-agent infrastructure consists of a system of reusable agent types that can be adapted to address a variety of different domain-specific problems. Each RETSINA agent draws upon a sophisticated reasoning architecture that consists of four reusable modules for communication, planning, scheduling, and execution monitoring.

A collection of RETSINA agents forms an open society of reusable entities, that self-organize and cooperate in response to task requirements. This agent infrastructure is currently being applied to a number of domains, including information retrieval, team decision making, and financial portfolio management.

RETSINA is a reusable agent. Each RETSINA agent has four reusable modules for communicating, planning, scheduling, and monitoring the execution of tasks and requests from other agents. A RETSINA agent is distinguished according to the kind of task it performs (i.e., interface, task, and information agents).

Zeus

Language

Java

Type

Agent building environment

Organization

British Telecommunications

URL

Zeus site

Functional Description

Zeus is a “collaborative” agent building environment and component library written in Java. Each ZEUS agent consists of a definition layer, an organizational layer, and a coordination layer. The definition layer represents the agent's reasoning and learning abilities; its goals, resources, skills, beliefs, preferences, etc. The organization layer describes the agent's relationships with other agents. The coordination layer describes the coordination and negotiation techniques the agent possesses. Communication protocols are built on top of the coordination layer and implement inter-agent communication. Beneath the definition layer is the API.

Commercial Systems

AgentBuilder

Language

Java

Type

Integrated agent and agency development environment

Organization

Reticular Systems, Inc.

URL

AgentBuilder site

Functional Description

AgentBuilder is an integrated tool suite for constructing intelligent software agents. AgentBuilder consists of two major components, the Toolkit and the Run-Time System. The AgentBuilder Toolkit includes tools for managing the agent-based software development process, analyzing the domain of agent operations, designing and developing networks of communicating agents, defining behaviors of individual agents, and debugging and testing agent software. The Run-Time System includes an agent engine that provides an environment for execution of agent software. Agents constructed using AgentBuilder communicate using the Knowledge Query and Manipulation Language (KQML) and support the performatives defined for KQML. In addition, AgentBuilder allows the developer to define new inter-agent communications commands that suit his particular needs.

All components of both the AgentBuilder Toolkit and the Run-Time System are implemented in Java. This means that agent development can be accomplished on any machine or operating system that supports Java and has a Java development environment. Likewise, the agents created with the AgentBuilder Toolkit are Java programs, so they can be executed on any Java virtual machine. Software developers can create powerful intelligent agents in Java that execute on a wide variety of computer platforms and operating systems. The AgentBuilder toolkit is designed to provide the agent software developer with an integrated environment for quickly and easily constructing intelligent agents and agent-based software.

Intelligent Agent Factory

Language

Java

Type

Agent development tool

Organization

Bits & Pixels

URL

Intelligent Agent Factory site

Functional Description

The Intelligent Agent Factory reduces the time needed to create intelligent agent solutions. The agents are intelligent in the sense that they are controlled with rules written in Jess (CLIPS), a forward-chaining system. Agents and rules are generated from simple specifications of workflows.

JACK Intelligent Agents

Language

JACK (extension of Java)

Type

Agent development environment

Organization

Agent Oriented Software Pty., Ltd

URL

JACK Intelligent Agents site

Functional Description

JACK Intelligent Agents is a third-generation agent system that provides the architecture and capability for developing and running software agents in distributed applications. Written in Java, it uses JACK Agent Language (an extension of Java) and thus provides all the benefits of the Java language.

JAM

Language

Type

Agent architecture

Organization

Intelligent Reasoning Systems

URL

JAM site

Functional Description

JAM is a Belief-Desire-Intention agent architecture based upon the Procedural Reasoning System (PRS) of Georgeff, Ingrand, Rao, Lansky, and others. Unlike many "agents" available today, that are useful in very restricted domains, JAM is applicable to nearly any application domain. It supports both top-down goal-based reasoning and bottom-up data-driven reasoning. Source code and documentation are available for download at the JAM website.

Jumping Beans

Language

Java

Type

Mobile components

Organization

Ad Astra Engineering, Inc.

URL

Jumping Beans site

Functional Description

Jumping Beans is a software framework that allows developers to mobilize a Java application so that it can move from one host to another during its lifetime. The mobilized software moves from host to host, along with the executable data, state, resources, and other essential information. This allows it to move to and execute on hosts that did not have the application previously installed.

LPA Agent Toolkit

Language

PROLOG

Type

Agent toolkit

Organization

Logic Programming Associates, Ltd

URL

LPA Agent Toolkit site

Functional Description

The LPA Agent Toolkit allows you to write agent-oriented programs using a logic-based approach. It does not impose any one-agent architecture, conversation policy, or an agent communication language, but provides the required building blocks to impleme nt any agent-based architecture you desire. This makes it flexible and suitable for prototyping, reading, and research purposes. Because agents are written in PROLOG, they can easily use the existing artificial intelligence and logic programming techniques. The Agent Toolkit provides a skeleton of an agent, which can exhibit the main agent characteristics, namely; autonomy, awareness, persistence, cooperation, and adaptiveness.

Madkit

Language

Java, Scheme, JESS

Type

Multi-agent development tool

Organization

Madkit Development Group

URL

Madkit site

Release Notes

Freely downloadable at http://www.madkit.org

Functional Description

Madkit is a publicly available multi-agent platform, based on organizational concepts. This platform is based on the organizationally based conceptual model called Agent/Group/Role (AGR), in which an agent is considered an active entity, which communicates by message passing with other agents and plays roles in groups. This organizational model eases the design and development of multi-agent systems and allows for the dynamic construction of applications. Madkit, which is written in Java, works in a distributed way across machines (with different Operating Systems) without needing a central server. Communications and groups can be freely distributed and the distribution is performed transparently at the application level. Agents can be programmed in multiple languages. For the moment Java, Scheme (Kawa), and JESS (a rule-based language written in Java and based on the CLIPS system) are the first available programming languages. Other languages will be available in the future. The Madkit architecture is totally modular. The micro-kernel (less than 60k of compiled Java code) is extended by a set of various libraries of messages, probes, and agents. System services and debugging tools are themselves provided as agents, making system extensions easy and simple to implement. The reduced size of the micro-kernel, combined with the principle of modular services, managed by agents, enable a range of multiple, scalable platforms, and construction of libraries of specialized agent models.

Microsoft Agent

Language

ActiveX

Type

Interface creatures

Organization

Microsoft

URL

Microsoft Agent site

Functional Description

Microsoft Agent is a set of programmable software services that supports the presentation of interactive animated characters within the Microsoft Windows interface. Developers can use characters as interactive assistants to introduce, guide, entertain, or otherwise enhance their Web pages or applications (in addition to the conventional use of windows, menus, and controls.). Microsoft Agent enables software developers and Web authors to incorporate a new form of user interaction, known as conversational interfaces, that leverages natural aspects of human social communication. In addition to mouse and keyboard input, Microsoft Agent includes optional support for speech recognition so applications can respond to voice commands.

NetStepper

Language

Type

Agent development environment

Organization

JW’s Software Gems

URL

NetStepper site

Functional Description

NetStepper allows you to design, test, and run Internet agents. By using 12 easy to learn action steps, you can program agents that interact with web pages, FTP sites, and even email servers. NetSteppers' powerful variable parsing and replacement make it easy to write scripts that massage web data and perform actions based on web page content.

OpenCybele

Language

Java

Type

Agent infrastructure

Organization

Intelligent Automation, Inc., MD

URL

OpenCybele site

Functional Description

OpenCybele is the open source release of the CYBELE™ agent infrastructure developed by Intelligent Automation Inc., Rockville, MD. It is a runtime environment built on top of the Java2™ platform for control and execution of agents. The agent execution is, however, event-driven and controlled by OpenCybele rather than by JVM. OpenCybele adopts a service-layered architecture, promoting plug-n-play capability of agent services, such as concurrency management, event management, thread management, internal event generation, communication (event generation & receive), and timer event generation. OpenCybele also promotes an Activity Centric Programming (ACP) paradigm for coding agents. The ACP paradigm encourages encapsulation of the agent's autonomous behavior as a group of activities. The activities can be coded using the core Activity Oriented Programming Interfaces (AOPIs) that are provided as a part of the OpenCybele kernel. The AOPIs, in turn, use the published interfaces of different agent services. Other features of OpenCybele include location independent communication, publish-subscribe based messaging, synchronous, asynchronous, broadcast, and point-to-point messaging.

UMPRS

Language

C++

Type

Agent Architecture

Organization

Intelligent Reasoning Systems

URL

UMPRS site

Functional Description

UMPRS is a belief-desire-intention agent architecture based upon the Procedural Reasoning System (PRS) of Georgeff, Ingrand, Rao, Lansky, and others. Unlike many "agents" available today, which are useful in very restricted domains, UMPRS is applicable to nearly any application domain. It supports top-down, goal-based reasoning and selects goals and plans based on maximal priority. Source code and documentation are available for download at the UMPRS website.

Voyager

Language

Java

Type

Agent-enhanced ORB

Organization

Object Space

URL

Voyager site

Functional Description

Voyager is a 100 percent Java agent-enhanced Object Request Broker (ORB). It combines the power of mobile autonomous agents and remote method invocation with complete CORBA support, and comes complete with distributed services, such as directory, persistence, and publish subscribe multicast. Voyager allows Java programmers to quickly and easily create sophisticated network applications using both traditional and agent-enhanced distributed programming techniques. Voyager uses regular Java message syntax to construct remote objects, send them messages, and move them between applications. Voyager allows agents (i.e., autonomous objects) to move themselves and continue executing as they move. In this way, agents can act independently on the behalf of a client, even if the client is disconnected or unavailable. This approach is particularly valuable in any type of workflow or resource automation.

Appendix C:
Expert System Inference Engines

Free/Cheap Expert System Shells

FOCL is an expert system shell and machine-learning program written in Common Lisp. The machine-learning program extends the Quinlan's FOIL program by containing a compatible explanation-based learning component. FOCL learns Horn Clause programs from examples and (optional) background knowledge. The expert system includes a backward-chaining rule interpreter and a graphical interface to the rule and fact base. For details on FOCL, see: Pazzani, M. and Kibler, D., "The role of prior knowledge in inductive learning," *Machine Learning* 9:54–97, 1992. It is available by anonymous FTP from ics.uci.edu:/pub/machine-learning-programs/ as the files README.FOCL-1-2-3, FOCL-1-2-3.cpt.hqx (a binhexed, compacted Macintosh application), FOCL-1-2-3.tar.Z (Common Lisp source code), and FOCL-1-2-3-manual.hqx (binhexed manual). If you use a copy of FOCL, or have any comments or questions, send email to pazzani@ics.uci.edu.

SOAR—ftp.cs.cmu.edu

/afs/cs.cmu.edu/project/soar/public/Soar5/ (Lisp Version)
/afs/cs.cmu.edu/project/soar/public/Soar6/ (C Version)

Contact soar-request@cs.cmu.edu Integrated Agent Architecture. Supports learning through chunking.

BABYLON is a development environment for expert systems. It includes frames, constraints, prolog-like logic formalism, and a description language for diagnostic applications. It is implemented in Common Lisp and has been ported to a wide range of hardware platforms. Available by anonymous FTP from ftp.gmd.de:/gmd/ai-research/Software/Babylon/ [129.26.8.84] as a BinHexed stuffit archive, on the Web via the URL http://www.gmd.de/, on the Apple CD-ROM, or with the book *The AI Workbench BABYLON*, which contains full source code of BABYLON and the stand-alone version for the Mac. The book describes the use of BABYLON in detail.

MOBAL is a system for developing operational models of application domains in a first-order logic representation. It integrates a manual knowledge acquisition and inspection environment, an inference engine, machine learning methods for automated knowledge acquisition, and a knowledge revision tool. By using MOBAL's knowledge acquisition environment, you can incrementally develop a model of your domain in terms of logical facts and rules. You can inspect the knowledge you have entered in text or graphics windows, augment the knowledge, or change it at any time. The built-in inference engine can immediately execute the rules you have entered to show you the consequences of your inputs, or answer queries about the current knowledge. MOBAL also builds dynamic sort taxonomy from your inputs. If you wish, you can use several machine learning methods to automatically discover additional rules based on the facts that you have entered, or to form new concepts. If there are contradictions in the knowledge base due to incorrect rules or facts, there is a knowledge revision tool to help you locate the problem and fix it. MOBAL (release 3.0b) is available free for non-commercial academic use by anonymous FTP from ftp.gmd.de:/gmd/mlt/Mobal/. The system runs on Sun SparcStations, SunOS 4.1, and includes a graphical interface implemented using Tcl/TK.

OPS5—ftp.cs.cmu.edu:/user/ai/areas/expert/systems/ops5/ops5.tar.gz

Micro Interpreter for Knowledge Engineering **(MIKE)** is a full-featured, free, and portable software environment designed for teaching purposes at the United Kingdom's Open University. It includes forward and backward chaining rules with user-definable conflict resolution strategies, and a frame representation language with inheritance and "demons" (code triggered by frame access or change), plus user-settable inheritance strategies. Automatic "how" explanations (proof histories) are provided for rule execution, as are user-specified "why" explanations. Coarse- and fine-grained rule tracing facilities are provided, along with a novel "rule graph" display which concisely shows the history of rule execution. MIKE, which forms the kernel of an Open University course on Knowledge Engineering, is written in a conservative and portable subset of Edinburgh-syntax Prolog, and is distributed as non-copy-protected source code. MIKE version 1 was described in the October/November 1990 issue of *BYTE*. MIKE v1.50, which was formerly available from a range of FTP servers, has been superseded by two newer versions: MIKEv2.03, a full Prolog source code version, incorporating a RETE algorithm for fast forward chaining, a truth maintenance system, uncertainty handling and hypothetical worlds, and MIKEv2.50, a turnkey DOS version with menu-driven interface and frame and rule-browsing tools, fully compatible with MIKEv2.03, but without source code. They are available by anonymous FTP from hcrl.open.ac.uk [137.108.81.16] as the files

MIKEv2.03: /pub/software/src/MIKEv2.03/*

MIKEv2.50: /pub/software/pc/MIKEV25.ZIP

They are also available from the CMU AI Repository. For further information, please contact Marc Eisenstadt, M.Eisenstadt@open.ac.uk, Human Cognition Research Lab, The Open University, Milton Keynes MK7 6AA, UK, phone +44 908-65-3149, fax +44 908-65-3169.

ES. The October/November 1990 issue of *BYTE* also described the ES expert system. ES supports backward/forward chaining, fuzzy set relations and explanation, and is a standalone executable for IBM-PCs. ES is available by anonymous FTP from ftp.uu.net:/pub/ai/expert-sys/ [192.48.96.9] as summers.tar.Z. Ftp.uu.net is mirrored on unix.hensa.ac.uk [129.12.21.7] under /pub/uunet/.

WindExS (Windows Expert System) is a fully functional Windows-based forward chaining expert system. Its modular architecture allows the user to substitute new modules, as required, to enhance the capabilities of the system. WindExS sports Natural Language Rule Processor, Inference Engine, File Manager, User Interface, Message Manager, and Knowledge Base modules. It supports forward chaining and graphical knowledge base representation. Write etoupin@aol.com for documentation and operational system.

RT-Expert is a shareware expert system that lets C programmers integrate expert systems rules into their C or C++ applications. RT-Expert consists of a rule-compiler that compiles rules into C code, and a library containing the rule execution engine. RT-Expert for DOS works with Borland Turbo C, Borland C++, and Microsoft C/C++ compilers. The personal edition is licensed for educational, research, and hobby use. Applications created with RT-Expert personal edition are not licensed for commercial purposes. Professional editions are available for commercial applications using DOS, Windows, and UNIX environments. RT-Expert is available by anonymous FTP from world.std.com:/vendors/rtis/rtexpert. For more information, write to Real-Time Intelligent Systems Corporation rtis@world.std.com.

Free/Cheap Expert System Shells: CLIPS and Related Systems

CLIPS 6.0 (C Language Integrated Production System) is an OPS-like forward chaining production system written in American National Standards Institute (ANSI) C by NASA. The CLIPS inference engine includes truth maintenance, dynamic rule addition, and customizable conflict resolution strategies. CLIPS, including the runtime version, is easily embeddable in other applications. CLIPS includes an object-oriented language called COOL (CLIPS Object-Oriented Language), which is directly integrated with the inference engine. CLIPS runs on many platforms, including IBM PC compatibles (including Windows 3.1 and MS-DOS 386 versions), Macintosh, VAX 11/780, Sun 3/260, and HP9000/500. CLIPS is available from COSMIC at a "nominal" fee (the MS-DOS/Windows 3.1 version is $350 for the software and $115 more for the documentation, with discounts for U.S. academic institutions; for update orders, $100 and $200, respectively) for unlimited copies with no royalties. (CLIPS is available free to NASA, USAF, and their contractors for use on NASA and USAF projects.) For more information, send email to service@cossack.cosmic.uga.edu, write COSMIC, University of Georgia, 382 East Broad Street, Athens, GA 30602, call 706-542-3265, or fax 706-542-4807. The CLIPS help desk phone number is 713-286-8919 (fax 713-286-4479/244-5698) and email address is stbprod@fdr.jsc.nasa.gov. (The address is STB Products Help Desk, LinCom Corporation, 1020 Bay Area Boulevard, #200, Houston, TX 77058-2628.)

DYNACLIPS (DYNAmic CLIPS Utilities) is a set of blackboard, dynamic knowledge exchange, and agent tools for CLIPS 5.1 and 6.0. It is implemented as a set of libraries that can be linked with CLIPS 5.1 or CLIPS 6.0. Versions 3.0 and 3.1 will work with either CLIPS 5.1 or CLIPS 6.0. Source code is not available. Agents use the blackboard to communicate with other intelligent agents in the framework. Each intelligent agent can send and receive facts, rules, and commands. Rules and facts are inserted and deleted dynamically while the agents are running. Knowledge can be transferred on a temporary or permanent basis. For more information, please contact Yilmaz Cengeloglu, PO Box 4142, Winter Park, FL 32793-4142, or send email to cengelog@escmail.orl.mmc.com, yil@engr.ucf.edu or 73313.775@compuserve.com. It is available from the CMU AI Repository in ftp.cs.cmu.edu:/user/ai/areas/expert/systems/clips/dyna/

AGENT_CLIPS is a multi-agent tool for MACINTOSH. Multiple copies of CLIPS run at the same time on MACINTOSH. Each Agent (CLIPS) can send CLIPS commands to other active agents at run time. AGENT_CLIPS handles incoming commands automatically. Command transfer also means that agents can exchange facts/rules at run time. This is a form of Knowledge Exchange among intelligent agents. AGENT_CLIPS does NOT use Blackboard Architecture. Library for AGENT_CLIPS is also included in this package. You can link this library with other CLIPS applications, such as fuzzyCLIPS. AGENT_CLIPS can be obtained by anonymous FTP from ftp.cs.cmu.edu:/user/ai/areas/expert/systems/clips/agent/. It is also available from Compuserve, AIEXPERT Forum, Libraries, and Expert System. For more information, contact Yilmaz Cengeloglu at 73313.775@compuserve.com.

FuzzyCLIPS 6.02 is a version of the CLIPS rule-based expert system shell with extensions for representing and manipulating fuzzy facts and rules. In addition to the CLIPS functionality, FuzzyCLIPS can deal with exact, fuzzy (or inexact), and combined reasoning; thereby allowing fuzzy and normal terms to be freely mixed in the rules and facts of an expert system. The system uses two basic inexact concepts, fuzziness and uncertainty. Versions are available for UNIX systems, Macintosh systems, and PC systems. There is no cost for the software, but please read the terms for use in the FuzzyCLIPS documentation. FuzzyCLIPS is available via WWW (World Wide Web). It can be accessed indirectly through the Knowledge Systems Lab Server using the URL http://ai.iit.nrc.ca/home_page.html or more directly by using the URL http://ai.iit.nrc.ca/fuzzy/fuzzy.html or by anonymous FTP from ai.iit.nrc.ca:/pub/fzclips/. For more information about FuzzyCLIPS, send email to fzclips@ai.iit.nrc.ca.

wxCLIPS provides a simple graphical front end to CLIPS 5.1, CLIPS 6.0 and CLIPS 6.0 with fuzzy extensions. It is essentially CLIPS modified to work with an event driven style of programming, and a set of GUI functions. wxCLIPS is available as Sun Open Look, Sun Motif, Linux Open Look, Windows 3.1, Windows 32-bit, and Windows 95 binaries. wxCLIPS is available by anonymous FTP from ftp.aiai.ed.ac.uk:/pub/packages/wxclips/ [192.41.104.6] or the URL http://www.aiai.ed.ac.uk/~jacs/wxclips/wxclips.html

Commercial Expert System Shells

ACQUIRE is a knowledge acquisition system and expert system shell. It is a complete development environment for building and maintaining knowledge-based applications. It provides a step-by-step methodology for knowledge engineering that allows the domain experts themselves to be directly involved in structuring and encoding the knowledge. (The direct involvement of the domain expert improves the quality, completeness, and accuracy of acquired knowledge; lowers development and maintenance costs; and increases their control over the form of the software application.) Features include a structured approach to knowledge acquisition; a model of knowledge acquisition based on pattern recognition; knowledge represented as objects, production rules, and decision tables; handling uncertainty by qualitative, non-numerical procedures; extremely thorough knowledge bases; sophisticated report writing facilities; and self-documenting knowledge bases in a hypertext environment. ACQUIRE-SDK, their Software Development Kit, provides callable libraries for MS-DOS, and SCO Unix; DLLs for Asmetrix ToolBook, Windows, Windows NT, Windows 95 and Win 32; and custom controls for Visual Basic. Call or email for information on how to utilize the SDK to deliver applications over the WWW. The ACQUIRE development package (knowledge acquisition system and expert system shell) costs $995 for Windows 3.1 and includes a manual, a tutorial, on-line help, and a telephone helpline. For more information, please visit their web page at http://vvv.com/ai/. For an example of an ACQUIRE application that is running over the web, try their Whale Watcher demonstration at http://vvv.com/ai/demos/whale.html For more information, write to Acquired Intelligence Inc, Suite 205, 1095 McKenzie Avenue, Victoria, Canada V8P 2L5, call 604-479-8646, fax 604-479-0764, or send email to <sales@aiinc.bc.ca>.

ACTIVATION FRAMEWORK runs on personal computers (DOS, Windows) and UNIX workstations. This tool is not a traditional expert system shell; rather it is a tool for building real-time data interpretation applications. The vendor claims the tool competes with Gensym's G2 in terms of application domains. For more information, write to the sales office at The Real-Time Intelligent Systems Corporation, 26 Worthen, Chelmsford, MA 01824, call 508-250-4633, or fax 508-256-8132. To reach the development office, write to 76 Otis Street, Westborough, MA 01581, call 508-870-0043, fax 508-870-0148, or send email to rtis@world.std.com.

Aion Development System (ADS) runs on numerous platforms, including DOS, OS/2, SunOS, Microsoft Windows, and VMS. It includes an object oriented knowledge representation, forward, backward, bidirectional, and pattern-matching rules, graphics, calls to/from other languages (C, Pascal, etc.), and the Choreographer graphical user interface. For more information, write to Aion Corporation, 101 University Avenue, Palo Alto, CA 94301, call 800-845-2466 (415-328-9595), or fax 415-321-7728. For Europe, write to Software Generation, Kontichsesteenweg 40, B 2630 Aartselaar, Belgium, call 32-(0)3-877.12.93, or fax 32-(0)3-877.13.55.

Angoss Knowledge Seeker. Angoss is a data-mining tool that can be used to produce knowledge bases of rules by inferring cause and effect relationships from a database. The DOS version costs $799 and the Windows version $899. For more information, write to Angoss Software International Ltd., 430 King Street W., Suite 201, Toronto M5V 1J5, Canada, call 416-593-1122, or fax 416-593-5077.

ART*Enterprise (Brightware, Inc., formerly a division of Inference Corporation). ART*Enterprise is the latest of the family of rule-based development environments, originating with ART in the mid-1980s. It is a development environment for enterprise-wide applications, incorporating rules, a full object system, which includes features currently not present in C++ or Smalltalk, and a large collection of object classes for user interface (UI) development across platforms (from Windows to OS/2 to UNIX), access to databases (SQL-based and ODBC-based), and multi-person development. The ART*Enterprise environment provides a forward chaining engine, where backward chaining can be implemented, though it is not supported directly. ART*Enterprise also provides a CBR kernel for those who are interested in incorporating it into their applications. For further information, contact Brightware, Inc., 101 Rowland Way, Suite 310, Novato, CA 94945, call 1-800-532-2890 (1-415-899-9070), fax 415-899-9080, or email info@brightware.com. Their home page is located at the URL http://www.brightware.com/

Arity Expert Development Package is an expert system that integrates rule-based and frame-based representations of knowledge with several different kinds of certainty factors. The OS/2 version costs $495 and the DOS version $295. For more information, write to Arity Corporation, Damonmill Square, Concord, MA 01742, call 800-722-7489 (508-371-1243), fax 508-371-1487, or email 73677.2614@compuserve.com or Paul G. Weiss, pgweiss@netcom.com.

BABYLON. For more information, write to VW-GEDAS GmbH, Pascalstrasse 11, W-1000 Berlin 10, call +49 30-39-970-0, or fax +49 30-39-970-999.

CAM Software sells two expert system tools, **DClass** and **LogicTree**. DClass is a decision-tree system used for manufacturing applications. LogicTree is a decision-making system designed for non-programmers. For more information, write to CAM Software, 390 W. 800 N., Suite 103, PO Box 276, Orem, UT 84059-0276, call 800-293-6777, or fax 801-225-0286.

CBR Express (Inference Corporation). CBR Express family of products supports case-based retrieval of information. For further information, contact Inference Corporation, 550 N. Continental Blvd., El Segundo, CA 90245, call 800-322-9923 (310-322-0200), fax 310-322-3242, or email marketing@inference.com. Their home page is located at the URL http://www.inference.com/

COGSYS. For more information, write to COGSYS Ltd., Enterprise House, Unit 37, Salford University Business Park, Salford M6 6AJ, England, or call 061-745-7604.

COMDALE/C, COMDALE/X, and ProcessVision. COMDALE/C is a real-time expert system designed for industrial process monitoring and control.

- COMDALE/C allows requests for justification of recommendations, conclusions, and control actions without interrupting the decision making process. It can deal with uncertainty in knowledge and data, and has an open architecture and time-based reasoning. Other features include: full object-oriented configuration; full networking capabilities; alarm processing; an interrupt driven controller; trending and historical data collection; time-scheduled events; a real-time database, and interfaces with Detector Control System (DCSs), Programmable Logic Controls (PLCs), and other I/O devices.
- COMDALE/X is an off-line consultative expert system, which queries the user for information required to make its decisions. COMDALE/X is included with COMDALE/C as the development tool for real-time expert systems. COMDALE/X has the capability to incorporate hypertext documents, with the reasoning abilities of the expert system, to produce expert hyper manuals, which provide information and generate advice through an easy to use interface.
- ProcessVision is a real-time process monitoring and control software package. Based on an open and modular architecture, ProcessVision provides a graphical operator interface; intuitive object-oriented display configuration, smart alarming, sensor validation, hot standby, and unlimited connectivity to all your process instrumentation in one global environment. For more information, write to Comdale Technologies (Canada) Inc., The Comdale Building, 701 Evans Avenue, Suite 600, Toronto, Ontario, CANADA, M9C 1A3, call 416-620-1234, fax 416-620-4526, or email info@comdale.com, or see their web site, http://www.comdale.com/.

C-PRS (Procedural Reasoning System in C) aims at representing and executing operating procedures. It allows the user to express and represent the conditional sequences of complex operations and to insure their execution in real time, while embedded in the application environment. C-PRS is useful for process control and supervision applications. The PRS technology has been applied to different tasks with real-time constraints and demands, including the monitoring of several subsystems of the NASA space shuttle, the diagnosis and supervision of telecommunication networks (Telecom Australia), the control of mobile robots (SRI, LAAS), the control system of surveillance aircrafts (Grumman) and air traffic management (Civil Aviation Authority). The procedural reasoning technology was initially developed at the Artificial Intelligence Center of the Stanford Research Institute (Menlo Park, California). ACS Technologies has further developed and now markets C-PRS, the first commercial implementation of the Procedural Reasoning technology. C-PRS was developed with the most currently recognized standards (i.e., C, UNIX, X11, MOTIF). It guarantees some real time properties. It is available on numerous platforms and operating systems, including SPARC, DECstation, Sony News, Hewlett Packard, VxWorks, and others. For more information, write to ACS Technologies, 5, Place du Village d'Entreprises, B.P. 556 31674 LABEGE Cedex, France. Call 33-62-24-99-20, fax 33-61-39-86-74, or email ingrand@ingenia.fr or cprs@ingenia.fr.

CPR (Case-based Problem Resolution) is a C++ class library, and Help!CPR is a helpdesk and knowledge authoring application that uses CPR. CPR is being embedded within call tracking systems and Help!CPR competes directly with Inference Corporation's CBR Express and CasePoint (CasePoint being CBR Express without authoring capabilities). For more information, write to The Haley Enterprise, Inc., 413 Orchard Street, Sewickley, PA 15143, call 800-233-2622 (412-741-6420), fax 412-741-6457, send email to info@haley.com, or see their web site http://www.haley.com/ or FTP site ftp://ftp.haley.com/. These sites include their literature and software in PostScript and HTML.

CRYSTAL runs on personal computers and is available from Intelligent Environments. For more information, write to Intelligent Environments Europe Ltd., Crystal House, PO Box 51, Sunbury-on-Thames, Middlesex TW16 7UL, England, call 44-0-932-772266, or fax 44-0-932-771499. See PC Magazine 8(2), January 31, 1989.

CxPERT is an expert system shell that produces royalty-free C code. For more information, write to Software Plus Ltd., 1315 Pleasant Meadow Road, Crofton, MD 21114, or call 301-261-0264.

The Easy Reasoner™ is a Case-Based Retrieval (CBR) tool with reasoning that provides adaptive associative memory; retrieves similar cases from memory given a new case; extends Query-by-Example (QBE) by providing Query-by-Similarity™ (QBS); indexes existing databases using decision trees; supports xBase, ODBC, and SQL databases; maximizes information while minimizing complexity; automatically filters noise to simplify decision trees; induces automatically or under explicit control; ranks and retrieves classified cases by similarity; efficiently retrieves similar cases from large databases; efficiently retrieves most similar case from large DBs; efficiently retrieves "next most similar" case; supports multiple decision tree indices per database; supports multiple decision trees per field; classifies new information using any decision tree; automatically or interactively classifies new cases; handles missing data and "don't know" responses; provides customizable thesaurus for nominal fields; provides adaptive, context-sensitive, default reasoning; provides adaptive estimation using decision trees; predicts or ranks values for nominal or ordinal fields; automatically learns N-dimensional similarity spaces; quickly retrieves closest case from similarity space; ranks cases by distance in similarity space; retrieves records where similarity depends on text; circumvents spelling problems with N-M-grams; recognizes various forms of English dictionary words; automatically determines information content per word; automatically determines thesaural information content. The Easy Reasoner 16 bit Windows Toolkit costs $249, The Easy Reasoner 32 bit Windows Toolkit costs

$499, The Easy Reasoner OS/2 Toolkit costs $499, and The Easy Reasoner UNIX X/Motif Toolkit costs $999. UNIX Versions available for Sun OS, Sun Solaris, HP UX, Data General, AIX. For more information, write to The Haley Enterprise, Inc., 413 Orchard Street, Sewickley, PA 15143, call 800-233-2622 (412-741-6420), fax 412-741-6457, email info@haley.com, or see their web site http://www.haley.com/ or FTP site ftp://ftp.haley.com/. These sites include their literature and software in PostScript and HTML.

ECLIPSE runs on personal computers (DOS, Windows). System V UNIX and POSIX versions are also available. The syntax is derived from Inference Corporations' ART and is compatible with NASA's CLIPS. Features include data-driven pattern matching, forward and backward chaining, truth maintenance, support for multiple goals, relational and object-oriented representations, and integration with dBase. For more information, write to The Haley Enterprise, Inc., 413 Orchard Street, Sewickley, PA 15143, call 800-233-2622 (412-741-6420), fax 412-741-6457, send email to info@haley.com, or see their web site http://www.haley.com/ or FTPsite ftp://ftp.haley.com/. These sites include their literature and software in PostScript and HTML. Also see *IEEE Computer*, February 1991, pages 4-12. The cost is $499 for the Eclipse 16 bit Windows Toolkit, $999 for Eclipse 32 bit Windows Toolkit or Eclipse OS/2 Toolkit, and $1,999 for Eclipse UNIX X/Motif Toolkit. Versions are available for Sun OS, Sun Solaris, HP UX, Data General, and AIX.

Emerald Empower Procedural Advisor for Macintosh or PC Windows. Costs $6800. Fault trees are built graphically to automate the decision path that an expert uses to troubleshoot a problem. Emerald Intelligence, Inc, 3850E Research Park Dr, Ann Arbor, MI 48108, call 313-663-8757, or fax 313-663-8757.

Esteem is a case-based reasoning tool for Windows that integrates case-based reasoning with rules. For more information, write to Esteem Software Inc., 302 E. Main, Cambridge City, IN 47327, call 317-478-3955, or fax 317-478-3550.

EXSYS Professional runs under MS-DOS, MS-Windows, Macintosh, SunOS, Solaris, UNIX, and Vax. It supports backward and forward chaining, linear programming, fuzzy logic, neural networks, and has a SQL interface. For more information please visit their web page at http://www.exsysinfo.com/. For examples of EXSYS applications running over the web, try their Web Runtime Engine demonstrations at http://www.exsysinfo.com/Wren/wren.html. For an MS-Windows executable demo of EXSYS Professional/EXSYS RuleBook, visit ftp://ftp.exsysinfo.com/pub/demos/.

These sites include their literature and software in PDF and HTML. For more information, send email to info@exsysinfo.com, write to Exsys, Inc., 1720 Louisiana Boulevard, NE, Suite 312, Albuquerque, NM 87110, call 800-676-8356 (505-256-8356), or fax 505-256-8359. Also see PC Tech Journal 7(1):115, January, 1989.

FLEX is a hybrid expert system toolkit available across a wide range of different hardware platforms. It offers frames, procedures and rules integrated within a logic-programming environment. FLEX supports interleaved forward and backward chaining, multiple inheritance, procedural attachment, and an automatic question and answer system. Rules, frames, and questions are described in English-like Knowledge Specification Language (KSL), which enables the development of easy-to-read and easy-to-maintain knowledge bases. FLEX is implemented in, and has access to, Prolog. FLEX is available from Logic Programming Associates (who originally developed FLEX on the PC), and also from most major Prolog vendors under license, including Quintus, BIM, Interface and ISL. FLEX has been used in numerous commercial expert systems, and prices on a PC running Windows or on a Macintosh start at around $1,000. (A review of Quintus-flex is expected in an upcoming issue of PC-AI. – mk.) For more information, contact: Logic Programming Associates Ltd, Studio 4, R.V.P.B., Trinity Road, London, SW18 3SX. Tel: +44 (0) 181-871-2016; Fax: +44 (0) 181-874-0449. Email: lpa@cix.compulink.co.uk. In the US, call 1-800-949-7567. Their web page is located at the URL http://www.lpa.co.uk

Foundation Technologies sells expert system products and services for financial applications, such as pension management. For more information, write to Foundation Technologies Inc., One Kendall Square, Cambridge, MA 02139, call 617-720-2760, or fax 617-720-4153.

Gensym's G2 offers a graphical, object-oriented environment for creating intelligent applications that monitor, diagnose, and control dynamic events in on-line and simulated environments. Featuring a structured natural language for creating rules, models, and procedures, G2 is the foundation of all Gensym application products and end-user applications. Gensym's application products include the G2 Diagnostic Assistant (GDA), which provides a visual programming environment for creating intelligent process management applications. NeurOn-Line, another Gensym product, allows users to easily create neural network applications. G2 includes concurrent execution of rules and procedures, and the ability to reason about behavior over time. G2 GUIDE allows users to easily create graphical end-user interfaces and real-time displays. Gensym's Telewindows provides a powerful multi-user client/server environment that allows users to share G2 applications. Gensym also offers G2 Bridge Products for connectivity to other programs (C and ADA) and real-time data systems including relational databases, distributed control systems, and programmable logic controllers. Gensym supports its products worldwide through 26 direct sales offices and a network of over 100 marketing partners. For more information, please write to Gensym Corporation, 125 Cambridge Park Drive, Cambridge, MA 02140 USA, call +617-547-2500, fax +617-547-1962, or send email to info@gensym.com. Gensym's homepage is located at http://www.gensym.com. (The Gensym page was formerly located at http://www.industry.net/gensym. The IndustryNet page will be removed this spring.)

GBB, generic blackboard, framework provides a high-performance blackboard database compiler and runtime library, which supports pattern-based, multidimensional range-searching algorithms for efficient proximity-based retrieval of blackboard objects.

- Knowledge Source (KS) representation languages
- Generic control shells and agenda-management utilities
- Interactive, graphic displays for monitoring and examining blackboard and control components.

These components provide the infrastructure needed to build blackboard-based applications. GBB is available for DOS/Windows, Mac, UNIX workstations (Sun, HP/Apollo, IBM, DEC, Silicon Graphics), Symbolics and TI Explorer Lisp machines. (GBB is a significantly enhanced, commercial version of the University of Massachusetts GBB research framework, available via

FTP from ftp.cs.umass.edu:/gbb/.) NetGBB, a distributed extension to GBB, provides the communication and coordination facilities GBB needs to build heterogeneous distributed blackboard applications. For more information, write to Blackboard Technology Group, Inc., 401 Main Street, Amherst, MA 01002, call 800-KSS-8990 or 413-256-8990, or fax 413-256-3179. To be added to the mailing lists, send email to gbb-user-request@bn.cs.umass.edu. There are two mailing lists, gbb-user (moderated) and gbb-users (unmoderated). Blackboard Technology has a World Wide Web page at the URL http://www.bbtech.com/.

GOLDWORKS III. For more information, write to Gold Hill Computers, Inc., 26 Landsdowne Street, Cambridge, MA 02139, call 800-242-5477 (617-621-3300), or fax 617-621-0656.

GURU is an expert system development environment and Relational Database Management System (RDBMS) that offers a wide variety of information processing tools combined with knowledge-based capabilities, such as forward chaining, backward chaining, mixed chaining, multi-value variables, and fuzzy reasoning. For more information about GURU as well as the other database engines, development tools and services offered by Micro Data Base Systems, please write to Micro Data Base Systems, Inc., 1305 Cumberland Avenue, P.O. Box 2438, West Lafayette, IN 47906-0438, call 800-445-MDBS/6327 (317-463-7200), fax 317-463-1234, or send email to info@mdbs.com.

HUGIN System is a software package for construction of model-based expert systems in domains characterized by inherent uncertainty. The HUGIN System contains an easy to use probability based deduction system, applicable to complex networks with cause-effect causal relations subject to uncertainty. The HUGIN System presents a novel development. The implementation is based on an improvement from the award winning work by Lauritzen and Spiegelhalter, *Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems*. The HUGIN Demonstration, for Sun Workstations and PC-Windows, is available for anonymous FTP from hugin.dk:/pub/README (130.225.63.15). The World Wide Web homepage for HUGIN is accessible by the URL http://hugin.dk/. For more information, write to HUGIN Expert A/S, Niels Jernes Vej 10, DK-9220 Aalborg O, Phone +45 9815 6644, Fax: +45 9815 8550, Email: info@hugin.dk.

Icarus is an expert systems development tool for PCs. It includes links to Lotus and dBASE files, forward and backward chaining, and Bayesian confidence factors. For more information, write to Icarus, 11300 Rockville Pike, Rockville, MD 02852-3088, call 301-881-9350, or fax 301-881-2542.

ILOG RULES is a high performance embeddable rule-based inference engine. It is a forward chaining tool, written in C++ (hence it is object-oriented and supports inheritance mechanisms) and is also provided as a C++ library. It runs virtually on any UNIX platform (e.g., HP97X0, Sun4, RS/6000, DecStations) as well as on PCs running DOS (with or without MS/Windows) or OS/2. It extends OPS/5 with nested premises (objects as values), rule packets (logical grouping of rules), a full Truth Maintenance System (TMS) for efficient non-monotonic reasoning, compilation of rules into C/C++ code, and an object oriented data-model in C++. ILOG RULES work directly on user objects, so interfacing is straightforward. C/C++ code may be included in rule conditions and actions. ILOG RULES is based on the fast XRETE implementation of the RETE algorithm developed by Thomson-CSF. For more information, contact ILOG, Inc., 2073 Landings Drive, Mountain View, CA 94043, tel 415-390-9000, fax 415-390-0946, e-mail info@ilog.com. European customers should contact ILOG SA, 9, rue de Verdun, BP 85, 94253 Gentilly CEDEX, France, tel +33 (1) 49-08-35-00, fax +33 (1) 49-08-35-10, e-mail info@ilog.fr. The URL of ILOG's web page is http://www.ilog.fr.

INTELLIGENCE COMPILER. For more information, write to Intelligence Ware, 9800 S. Sepulveda Blvd. Los Angeles, CA 90045-5228, call 213-417-8896, or fax 213-417-8897.

KDS is a case-based system that produces rules from cases. For more information, write to KDS Corporation, 934 Cornell Street, Wilmeete, IL 60091-1405, call 708-251-2621, or fax 708-251-6489.

KEE, ProKappa, and Kappa are expert system development packages that run on personal computers, workstations, and Lisp machines. Features include Asynchronous Transfer Mode (ATM), rule-based reasoning and OOP support. For more information, write to IntelliCorp, Inc., 1975 El Camino Real West, Suite 101, Mountain View, CA 94040-2216, call 415-965-5700/5500 or fax 415-965-5647. In Europe, call +44-344-305305. Also see *CACM* 31(4):382-401, April, 1988.

KES and SNAP run on personal computers (KES $4,000), workstations (KES $10,000, SNAP $40,000 on most platforms), minicomputers (KES $25,000), and IBM mainframes (KES $60,000). Although KES can be purchased separately, it is part of SNAP. For more information, write to Software Architecture and Engineering, Inc., 1600 Wilson Boulevard, Suite 500, Arlington, VA 22209, call 703-276-7910, or fax 703-284-3821. Write to Template Software, 13100 Worldgate Drive, Suite 340, Herndon, Virginia 22070, call 703-318-1000, or fax 703-318-7378.

Knowledge Craft is an expert-system development toolkit for scheduling, design and configuration applications. For more information, write to Carnegie Group, 5 PPG Place, Pittsburgh, PA 15222, call 800-284-3424 (412-642-6900), or fax 412-642-6906.

KnowledgeWorks from Harlequin runs on a variety of UNIX platforms, including Sun Sparc and clones (SunOS and Solaris), IBM RS/6000 (AIX), DEC MIPS (Ultrix), DEC Alpha (OSF/1), HP PA (HP-UX) and SGI (IRIX). It includes a CLOS-based object system, OPS compatible forward chainer (2500 firings/sec on a Sparc2), Prolog compatible backward chainer (40 KLIPS), graphical programming environment, user-defined conflict resolution, MetaRule Protocol for extending execution model and a SQL interface for relational databases. For further information, worldwide e-mail knowledgeworks-request@harlequin.com (or @harlequin.co.uk) or in the U.S., Fax 617-252-6505; Voice 800-WORKS-4-YOU (800-967-5749), 617-374-2400 or 617-252-0052. Mail to Harlequin Inc., One Cambridge Center, Cambridge, MA 02142 or in Europe, Fax 0223-872-519 (or 44-1223-872-519 from outside UK). Voice 0223-873-800 or 872-522 (or 44-1223-873-800 from outside UK). Telex 818440 harlqn g. Mail Harlequin Ltd., Barrington Hall, Barrington, Cambridge, CB2 5RG. Harlequin also sells LispWorks (a Common Lisp and Prolog programming environment described in part 4 of the Lisp FAQ), MLWorks (an ML programming environment) and Watson (an intelligence analysis tool).

K-Vision is a knowledge acquisition and visualization tool. It runs on Windows, DOS, and UNIX workstations. For more information, write to Ginesys Corporation, 1200 Woodruff Road, Suite C-9, Greenville, SC 29607-5731, call 800-277-8338 (803-288-8338), or fax 803-458-9050.

Laser. For more information, write to Bell Atlantic Knowledge Systems, Inc., P.O. Box 3528, Princeton, NJ 08543-3528, or call 800-552-2257 (609-275-8393).

LEVEL5 OBJECT for Microsoft Windows runs on an IBM PC compatible computer using Windows 3.1 or above. LEVEL5 OBJECT is a robust object-oriented application development system with a tunable inference engine product. For more information, write to Information Builders, 503 Fifth Avenue, Indialantic, FL 32903, call 800-444-4303 (407-729-9046), fax 407-727-7615, or email to 76366.323@compuserve.com, techsup@l5r.com, or sales@l5r.com. Their Web server is located at the URL http://www.l5r.com/level5.html. For Customer Support, call 407-984-8705.

There are several versions of the **M.4 product**. M.4 VB is a Visual Basic custom control that sells for $199. The full M.4 package runs on personal computers (DOS, Windows) and sells for $995. It features a modular kernel library that can be linked into C-language applications, backward and forward chaining, pattern matching, certainty factors, procedural control, an object-oriented representation, and ODBC hooks. M.4 is embeddable, configurable and extendable, and is provided as libraries, a custom control (VBX), and a DDE Server. Working VB, Visual C++, ToolBook, and DOS GUIs are provided to illustrate various integration techniques. M.4 is also available as embeddable libraries for Sun SPARCstations for $2,495 and as embeddable libraries for Macintosh System 7 for $995. For more information, write to Teknowledge Corporation (formerly Cimflex Teknowledge Corporation), 1810 Embarcadero Road, Palo Alto, CA 94303, or call 800-285-0500 (415-424-0500), fax 415-493-2645, or visit http://www.teknowledge.com/M4.

MailBot, from Daxtron Labs, is a personal E-Mail agent for Microsoft Mail. MailBot provides filtering, forwarding, notification, automatic question answering, and listserv-like capabilities. MailBot can act as an expert system shell for mail. The front-end translates user rule inputs into a variant of Prolog. An Application Programming Interface (API) exists for writing action code in Visual Basic. More information is available from the URL http://www.polaris.net/~daxtron/mailbot.htm.

MEM-1 is a Lisp-based language that aids in the development of Case-Based Reasoning systems; provides facilities to define case structures, create cases, divide cases into sub-cases, weigh case attributes, define indexing schemes and similarity functions, adapt solutions, and define procedural adaptation rules. Runs on Windows, DOS, SUN, DECStation, RS/6000, and Macintosh. Price: $199 ($50 for educational institutions). For more information, write to CECASE, 2291 Irving Hill Drive, Lawrence, KS 66045, call 913-864-4896, fax 913-864-7789, or send email to mem1info@cecase.ukans.edu.

ModelQuest 4.0. Runs on Windows 3+, NT, 95 ($995). ModelQuest is easy to learn and use, so you can quickly take advantage of its power. ModelQuest produces more accurate and robust models, as well as royalty-free C code. For more information, write to Abtech Corporation, 1575 State Farm Boulevard, Charlottesville, VA 22901, call 804-977-0686, fax 804-977-9615, e-mail sales@abtech.com, or visit their web page at http://www.abtech.com/.

MUSE. For more information, write to Cambridge Consultants, Science Park, Milton Road, Cambridge CB4 4DW, England, or call 0223-420024 Cambridge.

NEXPERT OBJECT runs on over 30 supported platforms, including personal computers ($5000), Macintosh ($5000), workstations ($12,000), minicomputers, and mainframes. Nexpert Object is written in C, and includes a graphical user interface, knowledge acquisition tools, and a forms system. For more information, write to Neuron Data, 156 University Avenue, Palo Alto, CA 94301, call 800-876-4900 (415-321-4488), or fax 415-321-3728. Other offices include New York, 212-832-8900; Philadelphia, 215-941-2981; Washington, 703-821-8800; Los Angeles, 714-851-4621; Chicago, 708-955-3688; Houston, 713-739-9020; United Kingdom, 44-71-408-2333, fax 44-71-495-6274; France, 33-1-40-70-04-21, fax 33-1-47-23-71-43; and Japan, 81-3-3746-4371, fax 81-3-3746-4374. See also *IEEE Software* 5(5):98, September, 1988; *PC Tech Journal* 6(11):112, November, 1988; *MacUser* 4(12):134, December, 1988; *MacUser* 4(9):136, September, 1989; *Computer Language* 6(12):123, December, 1989; *PC Week* 7(26):43, July 2, 1990; *MacWeek* 4(25):10, July 10, 1990; and *IEEE Expert*, December, 1991, page 72. (This entry is a bit out of date. Nexpert Object is now part of a larger product called "Smart Elements," which includes Open Interface 2.0—an object-oriented GUI developer—in addition to Nexpert Object 3.0.)

Object Management Workbench* (OMW) is the first object-oriented analysis and design tool, which incorporates directly executable diagrams and business rules. OMW works with Kappa (an integrated, visual programming environment for developing client/server applications.). Kappa applications are developed on UNIX and can be delivered on either UNIX or MS Windows. With Kappa CommManager*, an object request broker, customers can distribute Kappa applications, including program logic, across UNIX, MS Windows and MVS. Kappa-PC is IntelliCorps' integrated MS Windows-based programming environment for departmental developers. For more information, write to IntelliCorp, Inc., 1975 El Camino Real West, Mountain View, CA 94040-2216, call 415-965-5700 or fax 415-965-5647. In Europe, call +44-344-305305. See also *CACM* 31(4):382-401, April, 1988.

OPS83 was developed by the OPS5 developers as a successor to OPS5. OPS83 is written in C, and OPS83 rule bases can be embedded in C programs. OPS83 was the first OPS-like language to provide this integration of rule bases with C. OPS83 supports Generalized Forward Chaining (GFC), a new control structure that permits rules to be more expressive; one GFC rule can replace several conventional flat rules. (Details and examples are available on request.) The current version of OPS83 uses the proprietary RETE II algorithm to enable it to handle large, complex rule sets efficiently.

A multi-windowed, point-and-click development environment called the OPS83 Workbench is offered. OPS83 is available for DOS, OS/2, UNIX, VMS, and some proprietary operating systems. For information, write Production Systems Technologies, Inc., 5001 Baum Blvd., Pittsburgh, PA 15213, call 412-683-4000, or fax 412-683-6347.

Personal Consultant Plus. For more information, write to Texas Instruments, PO Box 2909, MS/2240, Austin, TX 78769, or call 800-527-3500.

PowerSMARTS (multimedia rule-based logic), **DataSMARTS** (database mining), **NeuroSMARTS** (neural net) for Macintosh or PC Windows. Costs range from $495 to $4995. Cognition Technology Corporation, 1000 Massachusetts Ave, Cambridge, MA 02138, call 617-492-0246, FAX 617-492-0247, Internet ctcma@aol.com.

RAL (Rule-extended Algorithmic Language) is a high-performance C-based expert system tool. It was designed to permit seamless integration of rules and objects into C programs. RAL is a superset of C. RAL rules can operate directly on the data types used by the C code in your application. RAL "understands" C type declarations, function prototypes, #define constants, etc. C expressions, statements, macros, function calls, etc. can be embedded directly in the rules. RAL has an open architecture that permits it to be used with any GUI builder, data base library, or other library that can be used with C. No bridges are required. For efficiency, RAL rules are compiled into C code. Charles Forgy, the inventor of RETE, has developed a more efficient match algorithm called RETE II for RAL. (Benchmark results are available on request.) A special version of the language, called RAL/RT, adds the features that are required for real-time expert systems. RAL and RAL/RT have a multi-windowed, point-and-click development environment called the RAL Workbench. RAL and RAL/RT are available for DOS, MS Windows, OS/2, and UNIX. License fees for development systems are available on request. There are no run-time fees charged for distributing applications developed using RAL. For information, write Production Systems Technologies, Inc., 5001 Baum Blvd., Pittsburgh, PA 15213, call 412-683-4000, or fax 412-683-6347.

ReMind. ReMind is a case-based reasoning tool. For more information, write to Cognitive Systems Inc., 880 Canal Street, Stamford, CT 06907, call 203-356-7756, or fax 203-356-7760.

Rete++ supports both forward and backward chaining. With Rete++, the programmer can develop object hierarchies and instantiate, manipulate, and access them using C++, the standard rule-based syntax, or C. Asserting and modifying the data, to be considered by rules, is done by creating a C++ instance, performing assignments and accessing the member data of the instance. Rete++ automatically generates C++ class taxonomies. The C++ components of Rete++ applications use these generated classes directly or further subclass them as needed. The Rete++ inference engine automatically considers any instances of a generated class (or its subclasses) in the matching of rule conditions. C++ data types provided by Rete++ allow more flexible representation and automatic reasoning, using standard C++ syntax without extraneous function calls. Rete++ automatically monitors changes to C++ objects without requiring programmers to explicitly code function calls. Rete++ is provided as a C++ class library. As such, it may be linked as part of your C++ application. You may completely embed Rete++, with or without its graphical development environment. Modules for developing Case Based Reasoning and integration, with databases, are available for Rete++ applications. Rete++ integrates a dependency based Truth Maintenance System to preserve logically sound and complete reasoning in spite of non-monotonicity. Multiple rule sets and agendas support modular development and cooperating expert systems. Rete++ has an advanced browser for rule-based programming. The Rete++ windowing development environment monitors the knowledge base, with multiple views, all updated in real-time, that allow for debugging, browsing, monitoring and setting of break points, and the tracing of rules, facts and goals. Rete++ 16 bit Windows Toolkit costs $999, Rete++ 32 bit Windows Toolkit costs $1,999, Rete++ OS/2 Toolkit costs $1,999, and Rete++ UNIX X/Motif Toolkit costs $2,999. UNIX Versions are available for Sun OS, Sun

Solaris, HP UX, Data General, AIX. For more information, write to The Haley Enterprise, Inc., 413 Orchard St., Sewickley, PA 15143, call 800-233-2622 (412-741-6420 or 412-967-1100), fax 412-741-6457, email info@haley.com, or see their web site http://www.haley.com/ or ftp site ftp://ftp.haley.com/. These sites include their literature and software in PostScript and HTML.

RT/Expert is an expert system development package, integrated into the SystemBuild development environment, that runs on PCs, UNIX, and VMS systems. For more information, write to Integrated Systems Inc., 3260 Jay Street, Santa Clara, CA 95054, call 408-980-1500, or fax 408-980-0400.

RTworks is a family of independent software modules, developed for intelligent real-time data acquisition and monitoring, data analysis, message/data distribution, and message/data display. RTworks offers a number of sophisticated problem-solving strategies, including knowledge-based systems, a point-and-click graphical user interface, temporal and statistical reasoning, and the ability to distribute an application over a heterogeneous network. Included with RTworks is a high-speed inference engine (RTie), which is used to analyze the data using objects, classes, procedures, and rules. The Inference Engine (IE) can perform trending, prediction and temporal reasoning of rapidly changing data. Displays can be built by non-programmers, using a user-friendly DRAW program. More than 60 different formats are provided for displaying input data, including strip charts, bar charts, control charts, dials, pie charts, and high-low graphs. Graphical objects can be tied to variables, which dynamically control attributes such as color, scale, rotation, motion, animation, and more. RTworks runs in a client-server architecture in which the RTserver process intelligently distributes the application's messages and data to only the client processes that need them. User-defined client processes can connect to the RTserver and send and receive messages with other processes in the application. Possible applications include process control, network monitoring, financial trading, and command and control. RTworks is available on a variety of UNIX and VMS platforms, under a floating license, in which you pay only for the number of simultaneous users, and the software is not node-locked to a particular machine. Current RTworks customers include Lockheed, NASA, Dow Chemical, PG&E (Pacific, Gas, and Electric), SWIFT, Mazda, and NTT. For further information, write to Talarian Corporation, 444 Castro Street, Suite 140, Mountain View, CA 94041, call 415-965-8050, fax 415-965-9077, e-mail info@talarian.com, or look on the worldwide web at http://www.mainstreet.net/talarian/.

SMECI is an expert system shell based on Lisp. For more information, contact ILOG, Inc., 2073 Landings Drive, Mountain View, CA 94043, tel 415-390-9000, fax 415-390-0946, email info@ilog.com. European customers should write to ILOG, 2, av. Gallieni, BP 85, 94253 Gentilly Cedex, France, tel +33 (1) 46-63-66-66, fax +33 (1) 46-63-15-82, email info@ilog.fr.

Sophos. For more information, write to Cognisys Consultants Inc., 7355 Trans Canada Highway, Suite 210, Saint-Laurent, Quebec, CANADA H4T 1T3, call 514-856-2311, or fax 514-856-2368.

STATUTE Corporate V3.0 for Windows. Includes facilities for automatic document generation, interfaces for Visual Basic, C, C++, and applications that can link to DLLs. Phone 1-800-229-1954 (616-242-1982). Fax 1-800-229-1959 (616-242-1948).

TechMate is a real life expert system designed to serve as a decision aid and productivity tool for maintenance engineers and technicians. TechMate minimizes the expertise required by the technician to troubleshoot and diagnose problems even in highly sophisticated and complex units. Oriented for the functional level, TechMate can troubleshoot units that contain analog, digital, and mechanical modules in electronic, electro-optic, hydraulic, or mechanical systems and devices. TechMate is used in many fault isolation applications in field service, depot facilities, and production lines. Its learning algorithms make TechMate grow smarter with use. TechMate is a model-based system that derives its intelligence from universal built-in knowledge bases and inference mechanisms, and from specific knowledge about the Unit Under Test (UUT) supplied by the test designer. It generates diagnostic assessments from the equipment's block diagram (which can be entered manually or imported electronically) and the characteristics of the symptoms and tests. It is also capable of diagnosing multiple faults. For a given set of symptoms and test results (including BIT results), TechMate's diagnostic algorithms rank candidate faulty modules by their likelihood of failure. It may also zoom in to further assess the fault likelihood of each individual sub-module. Next, TechMate identifies and evaluates the tests that may be used to isolate a fault, and proposes the most cost-efficient tests. The sequence of tests is dynamically reordered, based on the symptoms, BIT and test responses, thereby eliminating redundant tests. TechMate provides on-line documentation of the device and its tests. It also includes a database tool for storing and retrieving administrative data and sharing it with commercial DBMS. TechMate can be interfaced with a wide variety of test instruments and ATE systems. TechMate runs on Microsoft Windows and UNIX/X-Windows. TechMate customers include Texas Instruments, Northrop, Raytheon, Honeywell, BARCO, British Gas, Deutche Aerospace, and Siemens Plessey. For further information, call IET Intelligent Electronics at 617-229-5855, fax 617-221-5692, e-mail sales@ietusa.com, or look on the worldwide web at http://www.ietusa.com/.

TestBench, Shell is available from the Carnegie Group, Pittsburgh, Pennsylvania. The development environment runs on SUN workstations, and the production environment on a number of platforms including PCs and NeXT machines. For more information, write to Carnegie Group, 5 PPG Place, Pittsburgh, PA 15222, call 800-284-3424 (412-642-6900), or fax 412-642-6906.

VBXpert is a VBX custom control that uses the RETE Algorithm to add expertise to Visual Basic applications. VBXpert extends Visual Basic to support rule-based, knowledge-based systems development by encapsulating the RETE Algorithm within a VBX custom control. It is true custom control with Visual Basic property and event editing support, and provides convenient properties that can be accessed and set by VB code to control embedded reasoning. VBXpert also provides: inference engine events that execute code entered using Visual Basic editor; a general-purpose "Callback" event to call Visual Basic from the actions of rules; efficient processing, even for a thousand if-then rules; automatic sub-goaling and logical deduction; automatic retraction of invalid or outdated conclusions; Visual Basic declarations for Eclipse DLLs; support for Visual Basic's ANY and Variant data types; support for multiple concurrent Visual Basic apps; includes Eclipse DLLs and executable; includes C source code for the VBX. VBXpert for Windows $99 (price shown is for domestic check, VISA, MasterCard, and American Express orders only). The Haley Enterprise, Inc. 413 Orchard St., Sewickley, PA 15143-2029, phone (800) 233-2622 (412-741-6420), Info@Haley.COM, or see their web site http://www.haley.com/ or ftp site ftp://ftp.haley.com/.

Visual Expert is a GUI-based expert system development tool for Windows. For more information, write to SoftSell Technologies, 16150 NE 85th Avenue, Suite 224, Redmond, WA 98052, call 206-556-1436, or fax 206-883-9002.

VP-EXPERT version 3.1 runs on the IBM PC under DOS. It costs $349. A student version of the product is available for around $40. The student version is fully functional, but limited to 16k in the total size of the system. For more information, write to Wordtech Systems Inc., PO Box 1747, Orinda, CA 94563, call 510-689-1200, or fax 510-689-1263. Also, see *MacUser* 4(12):134, December, 1988.

XpertRule for Windows represents knowledge as decision trees, tables of decision examples, exception trees and sets of pattern rules. It can produce C code. Fuzzy Logic and Genetic Algorithm optimization are included. For more information, write to Attar Software USA, PO Box 68, Harvard, MA 01451-0068, call 800-456-3966, or fax 508-456-8383. European customers should write to Attar Software Limited, Newlands House, Leigh, Lancashire, UK WN7 4HN, or email info@attar.co.uk. Also see their home page at http://www.attar.com/.

YAPS is a tool for building expert systems and other programs that uses a rule-based knowledge representation in Lisp. The YAPS library provides a CLOS class, and appropriate methods, which the programmer may mix into his/her own classes or use directly. Rules and facts about an instance are associated with that instance. Instead of one large knowledge base with many rules, that is hard to debug and maintain, the programmer creates smaller knowledge bases, which are modular and more efficient. The YAPS knowledge bases can interact with and be controlled by the programmer's other modules, making hybrid systems straightforward. Introduced by Liz Allen at AAAI-83, YAPS is now available on Apple Macintosh, Sun3 and Sun4 (SPARC), DEC VAX under VMS and Ultrix, and 88Open platforms. On workstations, a single license costs $3995 and on the Macintosh (under Macintoch Common Lisp), it is $445. YAPS runs in most commercial Common Lisps, including Allegro CL, Harlequin LispWorks, Lucid CL, IBUKI CL, and Macintosh Common Lisp. YAPS is also available for the TI Explorer and Symbolic Lisp Machines, and a Flavors version is available for Sun3 in Franz Lisp. Other ports are underway; for price and availability contact College Park Software at 461 W. Loma Alta Dr., Altadena, CA 91001-3841; email info@cps.altadena.ca.us, or call 818-791-9153 (voice), or 818-791-1755 (fax).

Distribution

AIR UNIVERSITY LIBRARY
AIRFORCE RESEARCH LABORATORY (CODE 13)
ARMY MANAGEMENT STAFF COLLEGE LIBRARY
ARMY RESEARCH INSTITUTE LIBRARY
ARMY WAR COLLEGE LIBRARY
ASSISTANT DEPUTY CHIEF OF NAVAL OPERATIONS
ASN (M & RA)
CANADIAN DEFENSE LIAISON STAFF
CENTER FOR NAVAL ANALYSES LIBRARY
CHIEF OF NAVAL PERSONNEL (N-1, N00H (3), N00D, N1G1P, P-05, N13T1, N120C, N13WW)
COMMANDER NAVY PERSONNEL COMMAND (P00B, PERS-03, PERS-05, PERS-4, PERS-6, PERS-48, PERS-49,)
DEFENSE TECHNICAL INFORMATION CENTER
NAWCTSD
HEAD MANPOWER PERSONNEL TRAINING BRANCH (N813)
HUMAN RESOURCES DIRECTORATE TECHNICAL LIBRARY
JOINT FORCES STAFF COLLEGE LIBRARY
MARINE CORPS RESEARCH CENTER
MARINE CORPS UNIVERSITY LIBRARIES
NATIONAL DEFENSE UNIVERSITY LIBRARY
NAVAL HEALTH RESEARCH CENTER WILKINS BIOMEDICAL LIBRARY
NAVAL POSTGRADUATE SCHOOL DUDLEY KNOX LIBRARY
NAVAL RESEARCH LABORATORY RUTH HOOKER RESEARCH LIBRARY
NAVAL WAR COLLEGE LIBRARY
NAVY MANPOWER ANALYSIS CENTER
ONR (CODE 342)
OPNAV (N1, N1B, N10, N11, N12, N13)
PENTAGON LIBRARY
USAF ACADEMY LIBRARY
US COAST GUARD ACADEMY LIBRARY
US MERCHANT MARINE ACADEMY BLAND LIBRARY
US MILITARY ACADEMY AT WEST POINT LIBRARY
US NAVAL ACADEMY NIMITZ LIBRARY